Leggo VR perché mi dà la rotta

Il lettore di VR è giovane, dinamico, creativo. Di cultura e reddito superiore alla media, possiede spesso più di un videoregistratore, oltre all'impianto hi-fi e al computer: nel tempo libero, non rinuncia ai viaggi in Italia e all'estero, e a cinema, teatro e spettacoli sportivi in genere. Usa il videoregistratore non solo per i programmi tv o preincisi, ma anche per riprendere i momenti felici in famig[illegible] creare una videoteca personale.
E tu, che tipo di lettore sei?

VR

IMMAGINE MARSILII

Sommario

Amigames

Da pagina **41** a pagina **56** la consueta rassegna dei videogames in arrivo; ed i relativi commenti, severi come al solito!

Campus 64 / 128

Campus Amiga

Rubriche

Usa il tuo computer

Forse non tutti sanno che...

Dal mese di **luglio '90** verranno limitate le collaborazioni per il C/128.

Dal **gennaio '91** verrà evasa prevalentemente la corrispondenza pervenuta a mezzo BBS (modem).

Dal **gennaio '91** verranno privilegiate le collaborazioni che perverranno in Redazione a mezzo BBS.

Dal **gennaio '91** non verranno più affrontati argomenti relativi al C/128.

Dal **dicembre '91** non verranno più affrontati argomenti relativi al C/64.

Commodore Computer Club

COMMODORE COMPUTER CLUB

Direttore: Alessandro de Simone
[illegible]: Marco Miotti

Redazione - Collaboratori:
Davide Ardizzone - Claudio Baiocchi
Luigi Callegari - Umberto Colapicchioni
Donato De Luca - Carlo D'Ippolito
Valerio Ferri - Michele Maggi
Giancarlo Mariani - Domenico Pavone
Armando Sforzi - Dario Pistella
Fabio Sorgato- Valentino Spataro
Franco Rodella- Stefano Simonelli
Luca Viola

Grafica: Arturo Ciaglia

Redazione:
Via Mosè, 22 cap. 20090 OPERA (Mi)

Telefoni 02 / 57.60.63.10
Fax 02 / 57.60.30.39
BBS 02 / 52.49.211

Pubblicità:
Leandro Nencioni (dir. vendite)
Ketty Cusin
Via Mose', 22 20090 Opera (Mi)
tel. 02 / 57.60.63.10
Spazio Nuovo
Via P. Foscari, 70
cap. 00139 Roma
tel. 06 / 81.09.679

Edizioni:
Systems Editoriale s.r.l.
Via Mose, 22 20090 Opera (MI)
Reg. Naz. Stampa N. 01500
vol. 15 fg 793

Abbonamenti: Liliana Spina
Arretrati e s/w: Lucia Dominoni
Tel. 02 / 57.60.63.10

Tariffe: Prezzo per copia L. 6000
Abbonamento annuo (11 fascicoli) L. 60000
Estero: L. 100000 - Indirizzare versamenti a:
Systems Editoriale Srl c/c 37952207 oppure
inviare comune assegno bancario non
trasferibile e barrato due volte a:
Systems Editoriale Srl (servizio arretrati)
Via Mose', 22
cap. 20090 OPERA (Mi)

Composizione e fotolito:
Systems Editoriale
Stampa: La Litografica Srl Guggiono (MI)

Registrazioni: Tribunale di Milano
n. 370 del 2/10/82

Direttore Responsabile: Michele Di Pisa

Spedizioni in abbonamento postale gruppo
III. Pubblicita' inferiore al 70%

Distributore: Parrini - Milano

Pubblicazioni Systems:
Banca Oggi
Commodore Club (disco)
Commodore Computer Club
Commodore Computer Club
(disco, edizione tedesca)
Computer quotidiano
Electronic Mass Media Age
Hospital Management - Nursing '90
Personal Computer
Jonathan - TuttoGatto
Videoteca - VR Videoregistrare

Editoriale

Con la ripresa delle attività autunnali, di solito, si verifica un fermento sia tra gli operatori economici sia tra gli utenti.

C'è spazio per tutti, dai "professionisti" agli esperti, dai principianti agli sprovveduti e tutti sono presi dal demone della voglia del "nuovo", in accordo con il tradizionale desiderio comune di rinnovarsi dopo il periodo di vacanza estiva.

Coloro che non sono in grado di nascondere la propria trepidazione sono facile preda, come di consueto, di venditori senza scrupoli che, pur di vendere, sono sempre tentati dal trarre in inganno i probabili acquirenti evidenziando i pregi delle apparecchiature in vendita e nascondendone i difetti.

Al giorno d'oggi, fortunatamente, è (quasi) impossibile portarsi a casa un computer difettoso per natura, una stampante malfunzionante, un monitor dai colori sbiaditi. Se qualche difetto dovesse verificarsi, rientra nella normalità: non esiste una produzione esente da difetti e può succedere che qualche esemplare necessiti di una riparazione, una messa a punto, una sostituzione.

Ciò che, invece, non è tollerabile è l'intenzione di nascondere, all'utente meno informato, le *pecche di base* del sistema che, in quanto tale, si cerca di vendere.

Ad esempio, non tutti dicono all'ignaro acquirente che un computer Ms-Dos, privo di disco rigido, è praticamente inutilizzabile; alcuni, poi, trascurano di dire che certi elaboratori, per l'assenza di slot interni, sono espandibili solo a caro prezzo.

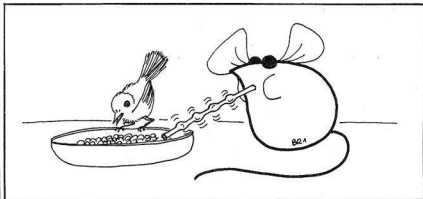
Chi, poi, desidera semplicemente un buon sistema per fare, quasi esclusivamente, del word processor, viene spesso invogliato all'acquisto di sistemi potenti e costosi che verranno sfruttati modestamente e, comunque, allo stesso modo di sistemi più semplici e, soprattutto, più economici.

C'è, infine, l'eterno discorso dell'assistenza: ci domandiamo, tra l'altro, se è legale una garanzia, che dovrebbe durare un anno, quando il centro di assistenza autorizzato trattiene l'apparecchio risultato difettoso per periodi che spesso raggiungono i tre mesi; e, per giunta, non sempre viene restituito funzionante.

Prima di effettuare un nuovo acquisto, quindi, consigliamo caldamente di esaminare con la massima attenzione ed obiettività le reali esigenze attuali e, per ciò che riguarda il futuro, almeno relative al prossimo triennio; inutile spingersi più in là.

Invitiamo anche a considerare con una punta di sospetto apparecchiature "professionali" offerte ad un prezzo troppo vicino al mezzo milione; rischiereste di portarvi a casa un computer senza Ram o senza drive o senza tastiera(!), per accorgervi troppo tardi che, sul depliant pubbliciario, era riportata in caratteri minuscoli la precisazione "*a partire* da L...".

LA VOSTRA POSTA

(a cura di A. de Simone)

Software rubato

Continuano a circolare opuscoli pubblicitari che pongono in vendita per corrispondenza, a caro prezzo, programmi e giochi di pubblico dominio, per di più piuttosto vecchi.

Invitiamo i lettori, prima di effettuare acquisti di software, ad entrare incontatto con appassionati o con banche dati (possibilmente gratuite, come la nostra) per ampliare il proprio parco - programmi in modo da evitare l'acquisto di antieconomici ed obsoleti pacchetti applicativi.

Amiga e C/64

E' possibile trasferire programmi dal C/64 ad Amiga?
(Fabio Tordi)

Il programma **C/64 emulator**, e la **scheda** che consente di connettere il drive 1541 ad Amiga, permette di realizzare quanto desideri.

L'operazione, tuttavia, è sconsigliabile per due motivi: anzitutto, sembra che possano verificarsi problemi seri in caso di malfunzionamenti (per disattenzione si rischia , addirittura, di bruciare la "porta" di Amiga).

Il motivo più importante, invece, è il secondo: a che serve far girare, su Amiga, i programmi del C/64? L'emulatore, infatti, poteva esseere interessante (ed, infatti, lo era) solo ai tempi in cui, per Amiga, era disponibile poco software.

Oggi, invece, il numero di programmi (e la qualità) per il nuovo computer della Commodore non fa rimpiangere nulla del buon vecchio 64.

Per non parlar del fatto che il software per Amiga ha raggiunto un prezzo paragonabile a quello del vecchio 8 bit...

Pochi sprite

Con il mio C/64 ho imparato a disegnare 8 sprite ma mi piacerebbe gestirne in numero maggiore. Come posso fare?
(Alessandro Zuppichini - Cormons)

Il C/64 può gestire un massimo di 8 sprite contemporaneamente. Nulla vieta, però, di avere in memoria decine di sprite e di far apparire quello che si desidera modificando, semplicemente, la locazione relativa all'ubicazione dello sprite in questione.

La locazione 2040 indica a partire da quale locazione si trovano i dati relativi allo sprite n. 0.

Se, ad esempio, in 2040 è presente il valore 13, i 63 valori necessari per tracciare lo sprite devono esser presenti a partire dalla locazione 13 * 64 (=832).

Per ovvi motivi, tuttavia, molti dei 256 valori pokabili non possono essere usati per non correre il rischio di interferire con locazioni di memoria vitali per il computer (puntatori, vettori, eccetera).

Con un po' di pazienza, tuttavia, è possibile individuare una zona, decisamente vasta, in cui allocare i vari sprite.

Quando si desidera selezionare un particolare sprite sarà quindi sufficiente pokare il valore opportuno per abilitarlo.

L'argomento, del resto già affrontato più di una volta, è comunque piuttosto ampio e non può essere affrontato in questa sede.

Ti consiglio di completare la raccolta di CCC o di sfruttare fino in fondo le nozioni apprese con il manuale in dotazione del tuo C/64.

VIDEOGAME,, CONSIGLI

Road Runner (pirata, C64)
All'inizio della partita, all'interno della grande roccia sul lato destro della strada, c'è una scorciatoia che permette di passare all'ultimo livello a cui si è arrivati con la partita precedente.

Around the world in 80 days (pirata, C64)
All'inizio di tutte le sezioni arcade (escluso la prima e l'ultima), si può premere Run / stop + restore e ottenere l'interruzione del gioco, visualizzando la consueta schermata vuota con Ready. A questo punto digitate Sys 2064 e, come per magia, la sezione verrà saltata fino al nuovo caricamento e al nuovo ritaglio di giornale. Fate attenzione a non spendere tutti i soldi, altrimenti il gioco si fermerà immediatamente.

Parsley (originale, C64)
Se si accede al data disk con un normale editor disco, si possono leggere tutte le domande con le relative risposte; la risposta esatta è quella contraddistinta da un più (+). Inoltre provate a scrivre *Galway* nella schermata di inizio (dove si trova il titolo del gioco che ondeggia)!

Daely tompson (pirata, C64)
Le scarpe adatte alla gara sono in fila, per esempio: per la corsa sono le prime, per il salto in alto le seconde ecc...

Dylan Dog (originale, C64)
I messaggi del disco schermate possono essere direttamente letti usando Easy Script: sono tutti file sequenziali!

The untouchables (pirata, C64)
Le schermate grafiche (i files che comiciano con PICTx) sono tutte in formato Koala, pertanto facilmente modificabili.
Per ridenominare un file che dovrà essere letto da Koala bisogna essere in possesso di un editor disco, andare nella traccia 18 e modificare i nomi come se fossero stati generati dal Koala. Questi sono sempre lunghi 15 caratteri (e non 16, come può sembrare).

Rocket ranger (tutte le versioni)
Nelle sezioni arcade, soprattutto contro gli aerei, è necessario spostarsi continuamente, senza fare economia di proiettili.
Nella sezione strategica è assolutamente antigienico lasciare un agente nello stesso territorio dopo una segnalazione: verrà sicuramente smascherato, con un antipatico messaggio da quell'idiota di Leermeister.

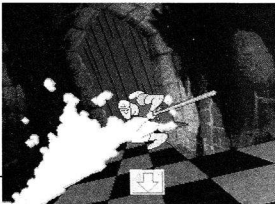
Combat school (pirata, C64 + speed-dos V9)
Si può caricare direttamente L e poi impartire SYS $4000: verrà caricata subito la seconda parte, con 2 players.

Batman - the movie (pirata, C64)
Nel secondo lato del disco basta impartire il comando:
Open 15, 8, 15, "c:a=c": close 15
In questo modo potrete ripartire sempre dalla cattedrale, basta non girare il disco, quando richiesto, e premere subito il fuoco.
Se non riuscite ad arrivare alla cattedrale, sostituire il comando di prima con:
Open 15, 8, 15, "c:a=b": close 15
In questo modo verrà sempre caricata la seconda sezione.
Il sistema non fa altro che ricopiare B o C sullo stesso disco col nome di A, per cui quando il programma carica A, verrà caricato rispettivamente B o C.

Consiglio generale
In molti giochi, che funzionano col multiload, è possibile passare subito dal primo schema all'ultimo ridenominando i loro nomi. Se, per esempio, il primo schema si chiama X1 e l'ultimo X5, basterà ridenominare X1 con X1- e poi chiamare X5 col nome X1. E' bene effettuare tale operazione su una copia di sicurezza.
Ringrazio i miei amici Giovanni C. e Mirko P.

Lista compilata da Marco Tonti (Mark)
Piazza Dossi, 6
47049 - Viserba (FO)
tel: 0541 / 734063

P.S: potreste togliemi una curiosità?
Con questa storia che gli articoli possono essere mandati solo su disco, che i trucchi devono essere mandati solo su disco, capisco che vi è indispensabile, ma quanti dischi riutilizzabili e riutilizzati avete lì? Qualche migliaio almeno! Scommetto che avete ricevuto anche dischi di qualità superiore: che cosa ne fate? Li buttate via? Trasferite i files su hard-disk e poi rivendete i rimanenti al miglior offerente?
Li raccogliete pazientemente uno ad uno e li archiviate in ordine alfabetico? Li rispedite al mittente?
Questa è solo una curiosità, non c'è nessuna allusione: pura curiosità.

Risposta: li cediamo gratuitamente alla Pia Opera *"Familiari ed amici dei collaboratori di C.C.C."*
Scherzi a parte, non hai idea di quanti soldi facciamo risparmiare agli aspiranti collaboratori che, seguendo il consiglio di telefonare prima dell'invio del materiale, si sentono respingere la propria offerta, benchè generosa.
Del resto, proprio per evitare spese (ma, soprattutto, per risparmiar tempo) invitiamo i lettori a servirsi della nostra banca dati evitando, di conseguenza, l'invio dei dischetti.

SYSTEMS EDITORIALE PER TE

La voce
Aggiunge al C/64 nuovi comandi Basic che consentono sia di far parlare il computer, sia di farlo Cantare! Diversi esempi allegati.
Cassetta: L. 12000 - Disco: L. 15000

Raffaello
Un programma completo per disegnare, a colori, con il C/64: linee, cerchi, quadrati, eccetera. Valido sia per disegno a mano libera che geometrico.
Cassetta: L. 10000

Oroscopo
Devi solo digitare la data di nascita e le coordinate geografiche del luogo che ti ha dato i natali. Vengono quindi elaborate le varie informazioni (case, influenze dei segni astrali, eccetera) e visualizzato un profilo del tuo carattere. Valido per qualsiasi anno, è indicato sia agli esperti sia ai meno introdotti. E' allegata una tabella delle coordinate delle più note città italiane e l'elenco delle ore legali in Italia dal 1916 al 1978.
Cassetta: L. 12000 - Disco: L. 12000

Computer Music
Cassetta contenente numerosi brani di successo da far eseguire, in interrupt, al tuo C/64 sfruttando, fino in fondo, il suo generatore sonoro (SID).
Cassetta: L. 12000

Gestione Familiare
Il più noto ed economico programma per controllare le spese e i guadagni di una famiglia.
Cassetta: L. 10000 - Disco: L. 10000

Banca Dati
Il più noto ed economico programma per gestire dati di qualsiasi natura.
Cassetta: L. 10000 - Disco: L. 10000

Matematica finanziaria
Un programma completo per la soluzione dei più frequenti problemi del settore.
Cassetta: L. 10000 - Disco: L. 20000

Analisi di bilancio
Uno strumento efficace per determinare con precisione i calcoli necessari ad un corretto bilancio.
Cassetta: L. 10000 - Disco: L. 20000

Corso di Basic
Confezione contenente quattro cassette per imparare velocemente le caratteristiche delle istruzioni Basic del C/64 e i rudimenti di programmazione. Interattivo.
Cassetta: L. 19000

Corso di Assembler
Un corso completo su cassetta per chi ha deciso di abbandonare il Basic del C/64 per addentrarsi nello studio delle potenzialità del microprocessore 6502. Interattivo.
Cassetta: L. 10000

Logo Systems
Il linguaggio più facile ed intuitivo esistente nel campo dell'informatica; ideale per far avvicinare i bambini al calcolatore.
Diversi esempi allegati.
Cassetta: L. 6500

Compilatore Grafico Matematico
Uno straordinario programma compilatore, di uso semplicissimo, che permette di tracciare, sul C/64, grafici matematici Hi-Res ad altissima velocità. Esempi d'uso allegati.
Cassetta: L. 8000

Emulatore Ms-Dos e Gw-Basic
Un prodotto, unico nel suo genere, che permette di usare, sul C/64 dotato di drive, la sintassi tipica del più diffuso sistema operativo del mondo. Ideale per studenti.
Solo su disco: L. 20000

Emulatore Turbo Pascal 64
Permette di usare le più importanti forme sintattiche del linguaggio Turbo Pascal (anche grafiche!) usando un semplice C/64 dotato di drive.
Ideale per studenti.
Disco: L. 19000

Speciale drive
Questo speciale fascicolo costituisce una guida di riferimento per le unità a disco del C64/128.
Comprende anche un velocissimo turbo-disk più la mappa completa della memoria del drive.
Fascicolo + disco: L. 12000

Utility 1
Un dischetto pieno zeppo di programmi speciali per chi opera frequentemente con il drive.
Disco: L. 12000

Utility 2
Seconda raccolta di utility indispensabili per realizzare sofisticate procedure di programmazione.
Disco: L. 15000

Graphic Expander 128
Per usare il C/128 (in modo 128 e su 80 colonne) in modo grafico Hi-res. Aggiunge nuove, potenti istruzioni Basic per disegnare in Hi-Res con la massima velocità in modalità 80 colonne.
Disco: L. 27000

Directory
Come è noto, a partire dal N. 10 di "Software Club" (la rivista su disco per l'utente dei "piccoli" computer Commodore), vengono riportati tutti i listati, in formato C/64-C/128, pubblicati su "Commodore Computer Club".
In precedenza tali listati venivano inseriti, mensilmente, in un dischetto, di nome "Directory", che oltre ai programmi di C.C.C. ospitava decine di altri file tra cui musiche nell'interrupt, giochi, listati inviati dai lettori e altro.
Ogni disco, dal prezzo irrisorio, contiene quindi una vera miniera di software. Ordinando i dischetti di "Directory" si tenga conto che al N. 1 corrispondeva il contenuto del N. 34 di "Commodore Computer Club", al N. 2 il N. 35 e così via.
Ogni dischetto: L. 10000

Super Tot '64
La nuova e completa edizione del programma Tot 13 con tutti i sistemi di riduzione e di condizionamento.
Ampia sezione dedicata alla teoria.
fascicolo + disco: L. 15000

Amiga Totospeed
Finalmente anche per Amiga un programma orientato alla compilazione delle schedine totocalcio.
Fai tredici conil tuo Amiga.
disco: L. 20000

Personal COMPUTER

Lire 6.000 La rivista per utenti di personal computer e workstation

Speciale Laptop

Quindici portatili a confronto

IN PROVA

Citizen Swift 24

Lotus Freelance Plus

Stress Manager

Professional File

IN PROVA

Il supercompatto

Flyer Base Station

IN PROVA

Norton Backup

Convivere con il Dos: Le espansioni di memoria

Personal Computer - Mensile d'informatica

SYSTEMS EDITORIALE PER TE

Disk'o'teca
Grazie a questa nutrita raccolta di brani musicali potrete divertirvi ascoltando i migliori brani prodotti dai vostri beniamini, oltre a una serie di composizioni prodotte "in casa".
In omaggio un bellissimo poster di Sting.
Disco: L.15.000

Assaggio di primavera
Esclusivo!
In un'unica confezione potrete trovare ben due cassette di videogiochi assieme a un comodo e funzionale joystick.
Cassette: L. 15.000

LIBRI TASCABILI

64 programmi per il C/64
Raccolta di programmi (giochi e utilità) semplici da digitare e da usare. Ideale per i principianti. **(126 pag.)**
L. 4800

I miei amici C/16 e Plus/4
Il volumetto, di facile apprendimento, rappresenta un vero e proprio mini-corso di Basic per i due computer Commodore. Numerosi programmi, di immediata digitazione, completano la parte teorica. **(127 pag.)**
L. 7000

62 programmi per C/16, Plus/4
Raccolta di numerosi programmi, molto brevi e semplici da digitare, per conoscere più a fondo il proprio elaboratore.
Ideale per i principianti. **(127 pag.)**
L. 6500

Micro Pascal 64
Descrizione accurata della sintassi usata dal linguaggio Pascal "classico". Completa il volume un programma di emulazione del PL/O sia in formato Microsoft sia in versione C/64 (da chiedere, a parte, su disco). **(125 pag.)**
L. 7000

Dal registratore al Drive
Esame accurato delle istruzioni relative alle due più popolari periferiche del C/64. Diversi programmi applicativi ed esempi d'uso. (94 pag.)
L. 7000

Il linguaggio Pascal
Esame approfondito della sintassi usata nel famoso compilatore. (112 pag.)
L. 5000

Simulazioni e test per la didattica
Raccolta di numerosi programmi che approfondiscono e tendono a completare la trattazione già affrontata sul precedente volume. **(127 pag.)**
L. 7000

Dizionario dell'Informatica
Dizionario inglese-italiano di tutti i termini usati nell'informatica. (Edizione completa). **(385 pag.)**
L. 10000

Word processing: istruzioni per l'uso
Raccolta delle principali istruzioni dei più diffusi programmi di w/p per i sistemi Ms-Dos: Word-Star, Samna, Multimate Advantage, Word 3. **(79 pag.)**
L. 5000

Unix
Un volumetto per saperne di più sul sistema operativo professionale per eccellenza.
Un necessario compendio per l'utente sia avanzato che inesperto **(91 pag.)**
L. 5000

ABBONAMENTO

Commodore Computer Club
11 fascicoli: L. 50.000

ARRETRATI

Ciascun numero arretrato di C.C.C. L. 6.000

Come richiedere i prodotti Systems

Coloro che desiderano procurarsi i prodotti della Systems Editoriale devono inviare, oltre alla cifra risultante dalla somma dei singoli prodotti, L. 3500 per spese di imballo e spedizione, oppure L. 6000 se si desidera la spedizione per mezzo raccomandata.
Le spese di imballo e spedizione sono a carico della Systems se ciascun ordine è pari ad almeno L. 50000.
Per gli ordini, compilare un normale modulo di C/C postale indirizzato a:

C/C Postale N. 37 95 22 07
Systems Editoriale Srl
Via Mosè, 22
20090 Opera (MI)

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento"), non solo il vostro nominativo completo di recapito telefonico, ma anche i prodotti desiderati ed il tipo di spedizione da effettuare.
Per sveltire la procedura di spedizione sarebbe opportuno inviare, a parte, una lettera riassuntiva dell'ordine effettuato, allegando una fotocopia della ricevuta del versamento.
Chi volesse ricevere più celermente la confezione deve inviare la somma richiesta mediante assegno circolare, oppure normale assegno bancario (non trasferibile o barrato due volte) intestato a:

Systems Editoriale
Milano

C'ERA UNA VOLTA IL CAOS. E POI FU SORT

Un argomento classico, ma sempre attuale, presentato attraverso alcuni brevissimi listati da digitare su un C/64-128 (dotato del nostro emulatore Gw-Basic) oppure su un qualsiasi Ms-Dos compatibile in grado di attivare il Gw-Basic originale Microsoft

di Giancarlo Mariani

Tempo fa fu presentato da C.C.C. un programma per Commodore 64, che consentiva a quest'ultimo di emulare il linguaggio basic dei computer IBM-compatibili (Gw-Basic).

L'emulatore, in pratica, consente di programmare sul C/64 "quasi" alla stessa maniera che su un computer Ms-Dos dotato di Gw-Basic.

Tale caratteristica ha permesso di sviluppare semplici programmi da digitare invariati su entrambe le macchine; hanno lo scopo sia di avvicinare l'utente di C/64 al mondo Ms-Dos, sia di far meglio comprendere alcune tecniche di programmazione rese possibili dall'interprete Gw-Basic.

L'argomento proposto in queste pagine è l'**ordinamento** alfabetico di un qualsivoglia numero di stringhe (o di numeri) che svilupperemo sfruttando due semplici programmi e discutendo sui vantaggi e svantaggi presenti nei più diffusi metodi di ordinamento.

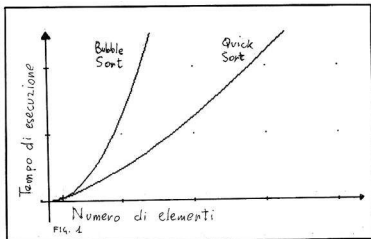



FIG. 1

PER CHI INIZIA

Il programma di queste pagine è destinato a chi si è procurato da poco tempo un computer Commodore. La sua lunghezza, infatti, è sufficientemente breve(!) da invogliare il lettore alla sua digitazione. Questa deve essere effettuata prestando la massima attenzione ai valori numerici presenti al suo interno; con particolare cura devono esser trascritti gli argomenti delle istruzioni, eventualmente presenti, Peek Poke, Sys. Anche se non riuscite a ben comprendere il reale significato delle varie istruzioni, non scoraggiatevi: digitate il tutto e registrate su supporto magnetico (comando save) il programma pubblicato.
Il valore del programma è anche didattico. E' infatti possibile rendersi conto delle varie difficoltà che si incontrano nello scrivere un listato, benchè semplice come quello che compare in queste pagine. Figuratevi, quindi, la pazienza che è necessaria per realizzare listati pieni di sprite, schermate in alta risoluzione, musica ed animazioni varie! A volte può capitare che il programma presenti malfunzionamenti a causa di una non corretta trascrizione del listato.
Nei casi più fortunati viene emessa una segnalazione di Syntax Error oppure di Illegal Quantity Error. In altri casi si rischia di bloccare il computer. C'è infine una terza eventualità: in caso di errore il programma gira e non segnala errori, ma l'elaborazione procede in maniera anomala ed erronea.
Si consiglia ancora, pertanto, di digitare il programma con la massima attenzione e di registrarlo sempre, prima di impartire il comando Run.

Il primo programma proposto (che, ripetiamo, si può digitare allo stesso modo sia sul Gw-Basic dei compatibili IBM che sul simulatore Gw-Basic 2 del C/64) si chiama **Ordina1** (vedi listato corrispondente) e consente di:

1) Caricare una lista di Cognomi - Nomi da disco.
2) Esaminare la lista
3) Ordinarla alfabeticamente
4) Salvarla su disco

La lista di cognomi - nomi è un semplice un file sequenziale, ogni riga del qua-

```
5 REM CREA.BAS
10 REM Crea il file "nomi.txt" e permette di inserire una lista di nomi
20 REM
100 KEY OFF: REM sul C64, KEYOFF (su Amiga: CLS)
130 CLS
140 PRINT "Scrivi il nome e il cognome"
150 PRINT "separati da uno spazio (* per finire) "
160 OPEN "nomi.txt" FOR OUTPUT AS #1
170 I = 1
180 PRINT "Numero "; I;: INPUT N$
190 IF N$ = "*" THEN GOTO 500
200 PRINT#1, N$
210 I = I + 1: GOTO 180
500 CLOSE 1
510 END
```

le contiene un cognome ed un nome nel seguente formato:

Cognome (spazio) Nome (ritorno carrello)

...in cui per (spazio) intendiamo la barra spaziatrice e per (ritorno carrello) il tasto Return, chr$(13).

Il brevissimo listato **Crea.Bas** permette di creare una lista "cavia" che consentirà di effettuare vari esperimenti.

Si provveda, quindi, a digitare il listato e a farlo girare, inserendo un numero di cognomi e nomi a vostro piacere. "Crea" li registrerà nomi su disco, in un file sequenziale chiamato **Nomi.Txt** destinato ad essere usato dal programma **Ordina1**.

Per quanto riguarda quest'ultimo, c'è da dire che dapprima apre in lettura il file **Nomi.Txt**, quindi "conta" le righe ivi presenti. Una volta contati i nomi, Ordina1 dimensiona una variabile stringa in modo tale da contenerli, quindi li carica. A caricamento avvenuto, potremo visualizzare la lista di nomi. A questo punto inizia l'ordinamento alfabetico, secondo uno dei metodi che vedremo dopo. Al termine Ordina1 chiederà se vogliamo esaminare la lista di nomi ordinata, se la vogliamo registrare su disco e con che nome.

Esaminando il listato vi sarete accorti che, così com'è, **non** può funzionare. Esso, infatti, ad un certo punto esegue un **Gosub 20000**, incaricato di richiamare la routine di ordinamento che invece... non c'è.

Infatti, per far funzionare correttamente il programma, dovremo digitare dalla riga 20000 in poi una delle routines di ordinamento proposte, che ora discuteremo. Ognuna di queste, infatti, accetta i parametri in ingresso allo stesso modo: alla variabile **C** viene associato il numero di elementi da ordinare, mentre all'array **N$ (1..C)** sono associati i cognomi-nomi da ordinare. Il risultato dell'ordinamento viene depositato nello stesso array N$(1..C).

Bubble Sort

Il listato **Bubble.Bas** elabora il metodo più conosciuto per ordinare un array. La sua popolarità deriva dal fatto che l'algoritmo di ordinamento è estremamente semplice e si basa soltanto su due cicli **For Next**.

Il metodo di ordinamento si basa su ripetuti confronti, ed eventuali scambi, di elementi adiacenti. Al primo ciclo l'elemento più grande viene "portato" nell'ultima posizione dell'array; al secondo ciclo, il secondo elemento più grande viene portato alla penultima posizione, e così via fino a raggiungere l'array ordinato. Dal momento che l'algoritmo del Bubble Sort è guidato da due cicli For, il numero di confronti (istruzione If) eseguiti per ordinare un array non dipenderà dallo stato iniziale dell'array stesso (ossia, se era completamente disordinato o già parzialmente ordinato), ma risulta, in ogni caso, uguale a...

0.5 * (n * n - n)

...in cui **n** è il numero di elementi da ordinare.

Il tempo impiegato dal sort per ordinare un array è proporzionale a **n x n**.

Da questo si può capire che bubble sort "crolla" quando gli elementi da ordinare sono numerosi.

Tutto sommato, conviene cercare un metodo più rapido.

```
5 REM ORDINA1.BAS
10 REM Carica il file " nomi.txt " , lo ordina alfabeticamente e quindi
20 REM permette di registrarlo
30 REM
100 KEY OFF: REM sul C64 (KEYOFF) ; su Amiga (CLS)
130 CLS
140 PRINT " Un momento, sto contando i nomi "
150 OPEN "nomi.txt" FOR INPUT AS #1
160 C = 0
170 IF EOF (1) < >0 THEN GOTO 300
180 INPUT #1,N$
190 C = C + 1: GOTO 170
300 PRINT " Il file 'nomi.txt' contiene " ;C; " nomi "
305 CLOSE 1
306 PRINT "Un momento, Sto caricando i nomi "
310 DIM N$ (C) : REM Variabile che dovra' contenere i nomi
320 OPEN "nomi.txt" FOR INPUT AS #1
330 FOR K = 1 TO C: INPUT#1, N$ (K): NEXT K: REM Carica i nomi in memoria
340 CLOSE 1
345 INPUT " Li vuoi vedere (s/n) " ;A$
346 IF A$ = " S " THEN GOSUB 1000
350 PRINT "Un momento, sto ordinando i nomi "
360 GOSUB 20000
370 INPUT " Vuoi vedere i nomi ordinati (s/n) " ;A$
380 IF A$ = " S " THEN GOSUB 1000
390 PRINT "Con che nome li vuoi registrare (CR per finire) "
400 N$ = "": INPUT N$: IF N$ = "" THEN GOTO 500
410 OPEN N$ FOR OUTPUT AS #1
420 FOR K = 1 TO C
430 PRINT #1, N$ (K)
440 NEXT K: CLOSE 1
500 END
1000 REM visualizza i nomi
1005 PRINT
1010 FOR K = 1 TO C: PRINT N$ (K): NEXT K
1015 PRINT
1020 RETURN
```

```
20000 REM SELECT.BAS: Ordinamento (SELECTION SORT)
20010 REM N$ (x) = Array da ordinare
20020 REM c = Numero di elementi da ordinare
20030 FOR A = 1 TO C - 1
20040 D = A
20050 A$ = N$ (A)
20060 FOR B = A + 1 TO C
20070 IF N$ (B) < A$ THEN D = B: A$ = N$ (B)
20080 NEXT B
20090 N$ (D) = N$ (A)
20100 N$ (A) = A$
20110 NEXT A
20120 RETURN
```

Selection Sort

Al listato **Select.Bas** è associato un metodo sicuramente migliore dal momento che, a differenza di Bubble Sort, confronta e scambia elementi adiacenti ed anche molto lontani tra loro, permettendo di ordinare in tempi brevi array particolarmente disordinati. Il funzionamento di selection sort è molto semplice: tra gli **n** elementi dell'array cerca il valore più basso e lo scambia con il primo elemento. In seguito esplora i rimanenti **n-1** elementi, cerca il valore più basso e lo scambia con il secondo elemento, e così via fino a quando ha esplorato, e quindi ordinato, l'intero array.

In pratica, è come avere un mazzo di carte da mettere in ordine "estraendo" ogni volta la carta più bassa e sistemandola in una pila a parte.

Una volta estratte tutte le carte, la pila risulterà ordinata. Sebbene il numero dei **confronti** effettuati da selection sort sia lo stesso di bubble sort, il numero degli **scambi** può essere molto minore (= minor tempo di elaborazione) in dipendenza delle condizioni iniziali di ordinamento dell'array.

Insertion Sort

Il listato **Insert.Bas** implementa il metodo **Insertion Sort**. Questo dapprima ordina i primi due elementi dell'array, poi, esplora l'array dal terzo elemento in poi. Il terzo elemento viene quindi inserito nella corretta posizione in relazione ai primi due elementi ordinati: il quarto viene inserito nella lista dei primi tre, e così via.

In pratica, riprendendo l'esempio del mazzo di carte, è come estrarre una carta alla volta per inserirla nella corretta posizione in un altro mazzo che si realizza a mano a mano.

Insertion Sort si comporta "naturalmente", ossia impiega poco tempo quando l'array inizialmente è già quasi ordinato, e lavora molto quando l'array è completamente disordinato.

Sebbene il numero di confronti da effettuare sia relativamente basso, ogni volta che bisogna inserire un elemento nella giusta posizione all'interno della lista già ordinata, la parte rimanente dell'array deve essere "shiftato" in avanti di una posizione.

Quest'ultima operazione rallenta notevolmente Selection Sort, che, comunque, può essere un'alternativa a insertion sort.

```
20000 REM INSERT.BAS: Ordinamento (INSERTION SORT)
20010 REM N$ (x) = Array da ordinare
20020 REM c = Numero di elementi da ordinare
20030 FOR A = 2 TO C
20040 A$ = N$ (A)
20050 B = A - 1
20060 WHILE B > 0 AND A$ < N$ (B)
20070 N$ (B + 1) = N$ (B)
20080 B = B - 1
20090 : WEND
20100 N$ (B + 1) = A$
20110 NEXT A
20120 RETURN
```

Shell Sort

Sebbene i tipi di ordinamento sinora visti siano validi, hanno il notevole difetto di richiedere un tempo di esecuzione proporzionale a **n x n**.

Questo particolare rende le procedure molto lente nel caso in cui il numero di elementi da ordinare sia molto alto. Per tale ragione sono stati sviluppati algoritmi che permettono di velocizzare notevolmente i compiti di ordinamento.

Con il metodo più noto (**Shell sort**) si esplora l'array usando un passo che ogni volta si riduce fino a diventare 1. Supponiamo, ad esempio, di avere 5 passi diversi: 9, 5, 3, 2,1. L'array verrà dapprima esplorato ed ordinato con passo 9, cioè il primo elemento sarà confrontato con il decimo, il decimo con il diciannovesimo, e così via.

Al secondo passaggio il passo diventerà 5 (primo elemento con il sesto, sesto con undicesimo... ecc.) .

```
20000 REM BUBBLE.BAS: Ordinamento (BUBBLE SORT)
20010 REM N$ (x) = Array da ordinare
20020 REM c = Numero di elementi da ordinare alfabeticamente
20030 FOR A = 2 TO C - 1
20040 FOR B = C TO A STEP - 1
20050 IF N$ (B - 1) > N$ (B) THEN A$ = N$ (B - 1): N$ (B - 1) = N$ (B): N$ (B) = A$: REM scambio
20060 NEXT B
20070 NEXT A
20080 RETURN
```

```
20000 REM SHELL.BAS: Ordinamento (SHELL SORT)
20010 REM N$ (x) = Array da ordinare
20020 REM c = Numero di elementi da ordinare
20030 A (1) = 9: A (2) = 5: A (3) = 3: A (4) = 2: A (5) = 1
20040 FOR W = 1 TO 5
20050 K = A (W)
20060 FOR I = K TO C - 1
20070 X$ = N$ (I + 1)
20080 J = I - K
20100 WHILE J > = 0 AND J < = C AND X$ < N$ (ABS (J + 1) )
20110 N$ (J + K + 1) = N$ (J + 1)
20120 J = J - K
20130 : WEND
20140 N$ (J + K + 1) = X$
20150 NEXT I
20160 NEXT W
20170 RETURN
```

Confronto tra i vari tipi di ordinamento

(tempi in secondi)	**C/64 + emulatore**	**Gw-Basic originale**	**IBM Quick-Basic**	**AmigaBasic**
Conta Nomi:	35	1.5	0.8	4
Carica Nomi:	32	0.93	0.8	3
Bubble Sort:	836	73	39.05	165
Select. Sort:	170	21.53	14.43	50
Insertion Sort:	381	33.07	15.77	69
Shell Sort:	214	13.18	7.03	28
Quick Sort:	120	60	30	76
Save nomi:	63	1.8	1.38	7

Il test è stato eseguito su **C/64** con drive **1541,** su computer **80286** IBM Compatibile con Hard Disk (ecco, quindi, giustificato il motivo dell'altissima velocità riscontrata nel colloquio con la periferica), e su **Amiga 500** dotato di drive standard. Sono state ordinate 200 stringhe composte da Cognome e Nome lunghe 41 caratteri ciascuna (20 caratteri cognome, spazio, 20 caratteri nome). I tempi di esecuzione sono espressi in secondi. Dal test si possono notare le differenze di velocità fra i vari tipi di sort e, soprattutto, l'enorme velocità dell'IBM e di Amiga nei confronti del C/64.

La prova è stata anche effettuata utilizzando, oltre all'interprete Gw-Basic dell'IBM, il compilatore **QuickBasic** della **Microsoft**. Si può notare il notevole incremento di velocità di quest'ultimo rispetto all'interprete Gw-Basic. Un'ultima nota: i tempi di ordinamento non devono essere presi come valori assoluti di confronto, dato che per la maggior parte degli algoritmi di sort proposti, lo stato iniziale di ordinamento dell'array ha un peso determinante sul tempo di esecuzione; inoltre il numero degli elementi presi in considerazione ("solo" 200) è troppo basso per permettere a routines del tipo **Shell Sort** o **Quick Sort** di evidenziare i notevoli vantaggi che le contraddistinguono dalle altre.

Alla fine dell'ultimo ciclo (con passo 1) saremo sicuri che l'array risulterà ordinato correttamente.

L'algoritmo è efficiente perchè ogni ciclo coinvolge un numero di elementi da ordinare relativamente basso, ed anche perchè, ad ogni ciclo, viene aumentato l'ordine dell'array. La sequenza dei passi (9, 5, 3, 2, 1) può essere cambiata per cercare di ottenere la migliore prestazione possibile. L'unica regola da osservare, per essere sicuri di ordinare l'array, è che l'ultimo passo deve essere 1. Il tempo di esecuzione di Shell Sort è proporzionale a **n exp (1/2)** dove **n** è il numero di elementi da ordinare.

In figura si vede che, all'aumentare del numero degli elementi, il tempo di esecuzione aumenta molto di più in Bubble Sort che non in Shell.

A proposito di sintassi

Vi sono alcune differenze di sintassi tra l'emulatore Gw-Basic per C/64 ed il Gw-Basic IBM.

I programmi proposti funzionano così come sono su entrambe le macchine, l'unica differenza è rappresentata dall'istruzione **Key Off**, che nell'IBM va digitata **Key (spazio) Off**, mentre l'emulatore del 64 richiede l'assenza dello spazio (**KeyOff**).

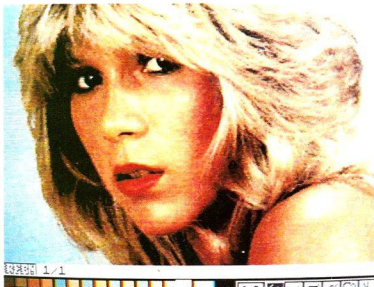
Un'altra lieve differenza sta nell'istruzione **Wend** (chiusura del ciclo **While**). Se viene digitata, sull'emulatore Gw-Basic del C/64, all'inizio di una riga, deve esser preceduta dal carattere di doppio punto (:), altrimenti non viene riconoscuta e viene emesso il messaggio **"While Without Wend"**.

Sui computer Ms-Dos compatibili, ovviamente, ciò non si verifica; tuttavia è consentito inserire caratteri di doppio punto in abbondanza, ed è per questo che, in tutti i listati di queste pagine, li potete notare in corrisponednza di istruzioni Wend ad inizio riga.

Quick Sort

QuickSort, come dice il nome, è uno dei più veloci metodi di ordinamento conosciuti. L'algoritmo dapprima sceglie un elemento *campione* dall'array da ordinare (che chiamiamo **X**), quindi divide l'array in due parti separando gli elementi minori di **X** da quelli maggiori.

Le due partizioni vengono a loro volta divise, scegliendo in ciascuna di esse un elemento campione e quindi mettendo gli elementi più piccoli da una parte della partizione e quelli più grandi dall'altra. Le quattro partizioni vengono divise ancora

```
20000 REM QUICK.BAS: Ordinamento (QUICK SORT)
20010 REM N$ (x) = Array da ordinare
20020 REM c = Numero di elementi da ordinare
20030 L = 1: R = C
20040 I = L: J = R
20050 X$ = N$ ( (L + R) /2)
20060 WHILE N$ (I) < X$ AND I < R
20070 I = I + 1
20080 : WEND
20090 WHILE X$ < N$ (J) AND J > L
20100 J = J - 1
20110 : WEND
20120 IF I > J THEN GOTO 20170
20130 Y$ = N$ (I)
20140 N$ (I) = N$ (J)
20150 N$ (J) = Y$
20160 I = I + 1: J = J - 1
20170 IF I < = J THEN GOTO 20060
20180 IF L < J THEN R = J: GOTO 20040
20185 R = C
20190 IF I < R THEN L = I: GOTO 20040
20200 RETURN
```

allo stesso modo, e così via fino a raggiungere l'array ordinato. L'elemento campione di ogni partizione può essere scelto in due modi: a caso, oppure prendendo il valore medio degli elementi della partizione.

Quest'ultimo è migliore e velocizza al massimo l'ordinamento, però impiega diverso tempo nella scelta dell'elemento campione.

La versione pubblicata in queste pagine (**QuickSort.Bas**) sceglie l'elemento campione nel mezzo della partizione, e, sebbene questa può non essere sempre la soluzione migliore, va benissimo nella maggior parte dei casi. La massima efficienza, con Quicksort, si può ottenere con una routine ricorsiva, ossia che richiama se stessa.

Il linguaggio interprete Gw-Basic non consente un'efficace ricorsività, pur se, in fin dei conti, la routine è sufficientemente veloce.

Un po' per volta

Passiamo ora ad esaminare un altro programma di ordinamento (**Ordina2.Bas**), anch'esso compatibile con **IBM** e **C/64-Gw-Basic,** che esegue l'ordinamento a mano a mano che l'array viene inserito da tastiera o da disco. Ordina2 presenta un menu tramite il quale è possibile scegliere tra 4 opzioni:

1 - Caricamento nomi : viene chiesto il nome del file nel quale è contenuta la lista di nomi, che vengono quindi caricati uno alla volta.

Si può notare che dopo aver caricato un nome (istruzione **Input#1, A$**), viene chiamata la routine alla riga 20000 (esaminata in seguito), che inserisce il nome nella corretta posizione all'interno dell'array **N$(1..C)**.

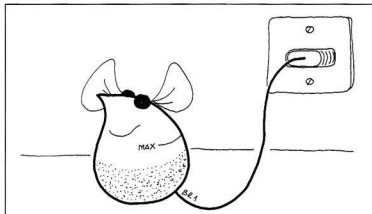
2- Aggiunta nomi : i nomi vengono inseriti da tastiera invece che da disco.

3- Salvataggio nomi : è possibile registrare la lista di nomi ordinata su disco, con un nome a scelta.

4- Visualizzazione nomi : permette di visualizzare la lista ordinata dei nomi inseriti sino a quel momento.

La routine di inserimento nomi (riga 20000...) funziona in pratica come un insertion sort. L'array contenente la lista di nomi inseriti è sempre **N$()**, mentre il nuovo nome da inserire è contenuto nella variabile **A$**. Dal momento che la routine viene richiamata ogni volta che si deve inserire un nuovo nome (da tastiera o da disco), N$() risulterà in qualsiasi momento ordinato correttamente. L'array N$() viene infatti esplorato dall'inizio alla fine allo scopo di individuare la corretta posizione, che chiameremo **p**, nella quale inserire il nuovo elemento A$ (la posizione, cioè, in cui A$ risulta maggiore di N$()). Tutti gli elementi posizionati "dopo" (da N$(p) in poi) vengono shiftati in avanti di un posto; in seguito N$(p) diventa A$. Il nuovo elemento è stato inserito!

Nella routine è stata aggiunta una piccola procedura che velocizza l'elaborazione: se l'elemento A$ da inserire è maggiore dell'ultimo elemento della lista N$(), viene messo automaticamente in ultima posizione, senza esplorare la par-

```
10 REM ORDINA2.BAS: Ordinamento di un vettore durante l'inserimento
20 REM
30 KEY OFF: REM sul C64: KEYOFF (su Amiga: CLS)
35 C = 0
36 DIM N$ (500): REM vettore che contiene i nomi
40 CLS
45 PRINT: PRINT "Nomi presenti in memoria: "; C: PRINT
50 PRINT "1 - Caricamento nomi"
55 PRINT "2 - Aggiunta nomi"
60 PRINT "3 - Salvataggio nomi"
65 PRINT "4 - Visualizzazione nomi"
70 PRINT "5 - Fine programma"
80 INPUT "Scelta: "; S$
90 IF S$ = "1" THEN GOSUB 1700
95 IF S$ = "3" THEN GOSUB 2000
96 IF S$ = "2" THEN GOSUB 4000
100 IF S$ = "4" THEN GOSUB 1900
110 IF S$ = "5" THEN CLOSE: END
120 GOTO 40
999 END
1700 REM Caricamento nomi
1702 CLS: PRINT "Che file vuoi caricare (* per finire) "
1703 INPUT N$: IF N$ = "*" THEN RETURN
1790 PRINT "Un momento, Sto caricando i nomi"
1800 C = 0
1805 OPEN N$ FOR INPUT AS #1
1807 IF EOF (1) < > 0 THEN GOTO 1820
1810 INPUT #1, A$: GOSUB 20000:
1811 REM Inserisce al posto giusto nella lista
1815 GOTO 1807
1820 CLOSE 1
1825 GOSUB 3000
1830 RETURN
1900 REM visualizza i nomi
1905 IF C = 0 THEN RETURN
1910 CLS
1920 FOR K = 1 TO C: PRINT N$ (K): NEXT K
1935 GOSUB 3000
1940 RETURN
2000 REM registra i nomi
2001 IF C = 0 THEN RETURN
2005 CLS: PRINT "Con che nome li vuoi registrare (* per finire) "
2010 INPUT N$: IF N$ = "*" THEN RETURN
2015 PRINT "Un momento, sto registrando i nomi"
2020 OPEN N$ FOR OUTPUT AS #1
2030 FOR K = 1 TO C
2040 PRINT #1, N$ (K)
2050 NEXT K: CLOSE 1
2055 GOSUB 3000
2060 RETURN
3000 REM aspetta un tasto
3010 PRINT: PRINT "Ho finito. Premi un tasto"
3020 A$ = INKEY$: IF A$ = "" THEN GOTO 3020
3030 RETURN
4000 REM Aggiunta manuale dei nomi
4010 CLS
4020 PRINT "Scrivi prima il cognome, poi spazio e poi il nome, e premi < CR > "
4030 PRINT " (* per finire) ": PRINT
4040 PRINT "Numero "; C + 1;: INPUT A$
4045 IF A$ = "*" THEN RETURN
4050 GOSUB 20000: REM Inserisce il nome nella lista
4060 GOTO 4040
20000 REM Ordina i nomi uno per volta
20010 REM N$ (x) = array contenente i nomi *
20015 REM A$ = nome da inserire nella lista *
20020 REM C = numero di nomi inseriti fino a quel momento *
20040 REM
20050 IF C = 0 THEN C = 1: N$ (1) = A$: RETURN
20060 IF A$ > N$ (C) THEN C = C + 1: N$ (C) = A$: RETURN: REM Velocizzazione
20070 K = 1
20080 IF K > C THEN GOTO 20100
20090 IF A$ > = N$ (K) THEN K = K + 1: GOTO 20080
20100 IF K > C THEN N$ (K) = A$: C = C + 1: RETURN
20110 FOR T = C TO K STEP - 1
20120 N$ (T + 1) = N$ (T)
20130 NEXT T
20140 N$ (K) = A$
20145 C = C + 1
20150 RETURN
```

te rimanente. Questo accorgimento consente di velocizzare notevolmente il caricamento da disco di sequenze di nomi già ordinate. I listati pubblicati possono essere estremamente utili ai fini della costruzione di programmi più ampi.

Ci si può sbizzarrire inventando nuove routines e confrontando le differenze di prestazioni tra macchine e procedure.

Chi possiede i due computer può digitare i programmi sul C/64 e, tramite collegamento via Rs-232 (vedi C.C.C. n. 49), trasferire i listati sul compatibile IBM.

L'occasioine consente inoltre, se mai ce ne fosse ancora bisogno, di porre in evidenza il notevole balzo di qualità che antepone il mondo Ms-Dos al pur validissimo C/64.

CAMPUS

64 / 128

SOMMARIO

18 - I GRAFICI INFINITI

Nulla riesce a stupire, in un personal computer, come la rappresentazione di grafici e di funzioni matematiche. Viene qui riproposto un emulatore del famoso **Spirografo**, il giocattolo "intelligente" che permette di realizzare un'infinità di disegni simmetrici. Le implementazioni presenti sono in ben tre linguaggi interpreti: C/64 Toma, C/64 Simon's Basic e **Amigabasic**. Con quest'ultimo computer, inutile dirlo, le velocità di elaborazione aumentano in modo considerevole.

23 - IL C/64 MISURA LUCE E CALORE

La connessione di un computer con il cosiddetto "mondo esterno" presenta aspetti davvero interessanti. Lo dimostra la semplice realizzazione di un nostro lettore che, con una manciata di componenti, suggerisce due applicazioni d'uso di un convertitore analogico-digitale. Per evitare danni al computer, si consiglia la costruzione della scheda solo a chi si ritiene realmente esperto in materia.

27 - C/64 IN ALTA RISOLUZIONE

Chi si avvicina per la prima volta alla grafica del C/64 annega nel mare di **Poke** e di **Peek** necessarie per gestire correttamente la pagina in alta risoluzione. Ecco quindi un prezioso articolo che chiarisce molti dubbi, anche se si lavora in Basic a velocità, come intuitivo, molto, molto bassa....

31 - C/128, UN RASTER TUTTO PER LUI

Molti si lamentano della mancanza di argomenti specifici per il C/128. Un nostro lettore presenta una tecnica di manipolazione del raster 128 davvero interessante.

Nessun albero viene abbattuto per gli inserti di Commodore Computer Club, stampati su carta riciclata al 100%

I GRAFICI INFINITI

Un paio formulette matematiche sono sufficienti per implementare uno dei più interessanti ed inesauribili procedimenti di tracciatura di grafici semi-casuali

di Roberto Morassi

I programmi di grafica sono, di solito, lunghi e complessi; cerchiamo di sfatare questa diceria

E' possibile tracciare, in alta risoluzione, le curve evolventi generate dal movimento di un punto **P**, interno ad un cerchio che rotola, senza strisciare, all'interno (o all'esterno) di una circonferenza fissa?

Limitiamoci, per semplicità, alla rotazione "interna", e supponiamo (vedi figura 1) che il centro **O** della circonferenza fissa sia all'origine di un sistema di assi cartesiani (**x, y**).

La coordinata X del punto P, in ogni istante, sarà data dalla somma di due componenti: la sua coordinata "locale" rispetto al punto C della ruota mobile, sommata alla coordinata di quest'ultimo rispetto allo zero. La stessa cosa vale per la coordinata Y.

L'angolo **B** (rotazione della ruota mobile su se stessa) e l'angolo **A** (rotazione del suo centro rispetto alla ruota fissa) sono, ovviamente, legati da un preciso rapporto matematico. Conoscendo il rapporto T tra i raggi delle due ruote si ricavano, con semplici regole di trigonometria, le seguenti relazioni:

T = R1 / R2

DC = R3 / R2 = R3 * T / R1

X = R1 * (1 -1/T) * cos (A) + (DC * R1/T) * cos (-A * (T - 1))

Y = R1 * (1 -1/T) * sin (A) + (DC * R1/T) * sin (-A * (T - 1))

Il rapporto può assumere qualunque valore reale, ma dato che in pratica si utilizzano delle

Repetita iuvant

Sul n. **30**(!) di C.C.C. fu proposto un problema di grafica con il quale si sfidavano i lettori a scrivere un programma che fosse in grado di simulare i meravigliosi disegni prodotti dallo **spirografo**.

L'autore dei due programmi (per C/64), purtroppo, rispose all'appello con un certo ritardo ed il suo lavoro fu accantonato.

La notevole brevità dei listati, e la successiva implementazione anche su **Amiga**, ci consente oggi di riprendere un argomento decisamente interessante e di pubblicare tre routinette (in Basic) che possono essere facilmente inserite, ad esempio, come presentazione di programmi matematici.

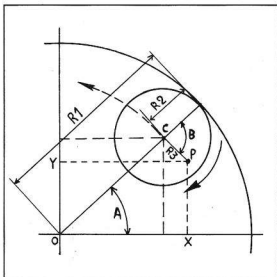



Figura 1
Schematizzazione dei dati che entrano in gioco per implementare la procedura dello spirografo descritta nell'articolo

ruote e corone dentate, è conveniente esprimere **T** come rapporto razionale fra il numero di denti della ruota fissa e quello mobile.

I programmi per C/64

I listati per il Commodore 64 sono due. Il primo richiede il preventivo caricamento (ed attivazione) delle **routines di Toma**; il secondo, invece, richiede la presenza dell'interprete **Simon's Basic**.

Tali linguaggi sono indispensabili per tracciare grafici dal momento che, come è noto, il C/64 è sprovvisto di istruzioni specifiche per la grafica. I listati (compreso quello per **Amiga**) richiedono in Input vari parametri:

numero di denti assegnati alla **ruota fissa**;

numero di denti della **ruota mobile** (con RF% maggiore di RM%);

decentramento DC del punto P, che va indicato come valore percentuale compreso tra **0** (cui corrisponde **P** al centro della ruota mobile, e in questo caso verrà tracciata solo una circonferenza) e **1** (corrispondente al punto P posizionato sul bordo).

```
' Spirografo per AmigaBasic

' di Roberto Morassi (adattamento Amiga by A. de Simone)
DIM r, z, a, t1, d, x, y, t, mx, x1, y1, rf%, rm%, dc, a$, s$, v, g$, p$
DEF FNx(x) = 320 + (r * COS (x + d) + z * COS (-x * t1 + d)) * 1.9
' Il valore 1.9 consente di visualizzare cerchi... rotondi e non ovali
DEF FNy(x) = 100 -(r * SIN (x+d) + z * SIN (-x * t1 + d))
pi = 3.1415: REM pigreco
PRINT "esempi di risposte:"
PRINT "n, 90, 50, 0.3, 1.05, f, n"
PRINT "n, 120, 28, 0.5, 1.01, f, s"
PRINT "s, 90, 200, 12, 0.8, 1.1, f, s"
PRINT "n, 10, 7, 0.5, 1, f, n"
PRINT "n, 137, 100, 0.6, 1, f, s"
PRINT  "s, 90, 351, 167, 0.5, 1.1, v, s"
PRINT  "s, 360, 100, 77, 0.7, 1.02, f, n"
PRINT  "s, 360, 100, 77, 0.7, 1, v, s"
PRINT: INPUT "rotazione asse (s/n)"; a$
IF a$ <> "s" THEN d = pi/2: GOTO RuotaFissa
INPUT "angolo di rotazione";d: d = pi / 2 -d * pi / 180
RuotaFissa:
INPUT "n. denti ruota fissa"; rf%
INPUT "n. denti ruota mobile"; rm%
t = rf% / rm% : t1 = t -1: IF t <= 1 THEN RuotaFissa
Decentramento:
INPUT "Decentramento";dc: z = dc * 90 / t
IF dc>1 THEN Decentramento
INPUT "Ampiezza linea (1 - 1.1)";Amp: IF Amp=0 THEN Amp=1.1
mx = rf% : FOR x=1 TO rf%
IF x * rf% / rm% = INT (x * rf% / rm%) THEN mx = x * rf% / rm%: x = rf%
NEXT: mx = 2 * pi * mx / t: r = 90 * (1 -1 / t)
INPUT "incremento fisso o variabile (f/v)"; a$
v = mx / 400: IF a$ = "f" THEN v = pi / 80
INPUT "cambio colore (s/n)"; Ccol$
Ccol=0: IF UCASE$ (Ccol$) = "S" THEN Ccol=1
col=1: x1 = (320 + (z + r) * COS (d)): y1 = 100 -((z + r) * SIN(d))
FOR a=0 TO mx STEP v: x = FNx(a): y = FNy(a)
LINE (x1 * Amp, y1 * Amp) - (x, y), col: x1 = x: y1 = y
IF Ccol=1 THEN col = col + 1: IF col > 3 THEN col=1
NEXT
```

La versione AmigaBasic dello spirografo.

Con i programmi grafici i computer sono messi a dura prova; lo dimostrano i tempi di elaborazione

Il Basic 2.0 del C/64 non offre comandi grafici; è quindi necessario ricorrere ad altri linguaggi interpreti

Nelle righe 260 / 270 viene calcolato il numero totale di **rivoluzioni** (MX, espresso in unità 2 * pigreco) necessario per riportare P al punto di partenza e "chiudere" la curva.

L'incremento dell'angolo A fra un punto e l'altro della curva può essere variabile o fisso, a scelta. Con il primo, MX viene diviso in un numero fisso di parti (**400**) e quindi le curve verranno tracciate tutte nello stesso tempo (circa 1 minuto e 15 secondi con il C/64; il tempo è sensibilmente minore con **Amiga**). Questo è il metodo più veloce, ma anche il meno preciso dal momento che la precisione sarà tanto minore quanto più lunga è la curva (vedi esempi di grafici riprodotti).

Con l'incremento fisso (pigreco / 80) il tracciamento sarà più preciso, ma inevitabilmente più lento. Per curve brevi, tuttavia, il grado di precisione è all'incirca lo stesso ed il secondo metodo risulta più veloce del primo.

In pratica conviene usare l'incremento variabile per un primo esame della curva, salvo ripeterla con incremento fisso se si vuole un disegno più "bello", o magari per stamparlo.

L'intero ciclo di tracciamento è svolto nella riga 320; dopo ogni incremento, il nuovo punto viene raccordato al precedente mediante l'istruzione **Draw**.

Una volta completato il disegno, si preme un tasto qualunque per ritornare all'inizio del programma.

Le curve successive possono, a scelta, essere sovrapposte alle precedenti per creare disegni di notevole simmetria e bellezza. All'inizio del ciclo il punto P si trova in alto, al centro dello schermo, sulla stessa verticale dei centri delle due ruote (asse principale della curva).

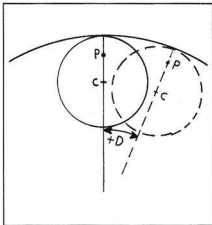



Figura 2
Significato grafico del termine D citato nell'articolo.

Rispondendo "**S**" alla richiesta "rotazione asse?" è possibile ruotare l'asse principale di un angolo **D** a scelta (espresso in gradi, vedi figura 2) e creare varianti ancora più complesse.

Iniziando

Per iniziare, provate a combinare alcuni valori di RF% e RM% fra quelli suggeriti qui di seguito; il resto... è affidato alla vostra fantasia. Si sottolinea che se il numero di denti della ruota grande è un multiplo di quella piccola (esempio: 80 e 20), il disegno ha molte probabilità di terminare dopo poche tracciature. Un rapporto tra numero di denti non divisibili tra loro (127 e 42, ad esempio) genera invece numerose curve prima che il programma termini.

Denti ruota **fissa**: 96, 105

Ruota **mobile**: 24, 30, 32, 36, 40, 42, 45, 48, 50, 52, 56, 60, 63, 64, 72, 75, 80, 84.

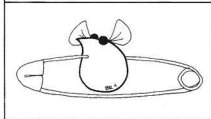
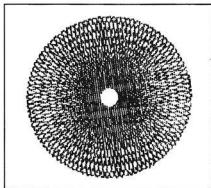
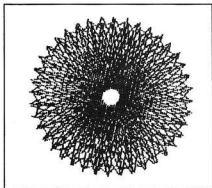
La versione Amiga

Grazie alla maggiore velocità offerta dai coprocessori grafici di Amiga è possibile aggiungere altre istruzioni in grado di offrire una varietà di opzioni davvero infinite senza allungare di molto i tempi di elaborazione.

Oltre alle domande sull'eventuale rotazione dell'asse, sul numero dei denti della ruota fissa e mobile e sul decentramento e sull'incremento fisso o variabile, compare una domanda relativa all'**ampiezza di linea.**

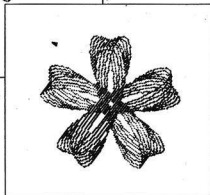
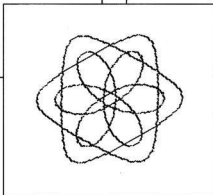
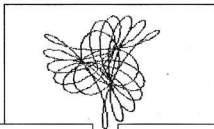
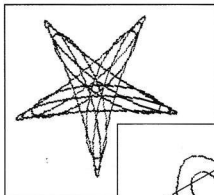
Rispondendo con **1** si ottiene un disegno simile a quello che si può ottenere con il C/64. Con valori maggiori, invece, è come se si utilizzasse un pennello, di larghezza proporzionale al valore digitato, che traccia quindi linee di spessore maggiore; si sconsiglia di superare il valore 1.5 per evitare di creare effetti sgradevoli a vedersi.

La domanda relativa al **cambio di colore** consente di ottenere la tracciatura di segmenti di raccordo il cui colore cambia continuamente (tra i tre disponibili su Amiga, bianco, nero e arancione: il blu, eguale al colore di fondo, viene evitato per ovvi motivi). Usando schermi con un maggior numero di colori sarà possibile, ovviamente, modificare l'effetto cromatico finale. Si noti che la funzione **DEF FNx(x)** presenta il coefficiente **1.9** (posto alla fine della formula)

```
SPIROGRAF    (ROUTINES TOMA)

130 IFPEEK(49890)=84ANDPEEK(49891)=79ANDPEEK(49892)=77THEN150
140 LOAD"GRAF/M",8,1
150 SYS51000:SYS49274
160 :
170 DIM R,Z,A,T1,D,X,Y,T,MX,X1,Y1,RF%,RM%,DC,A$,S$,V
180 DEF FNX(X)=R*COS(X+D)+Z*COS(-X*T1+D):DEF FNY(X)=R*SIN(X+D)+Z*SIN(-X*T1+D)
190 ←TEXT6,14:PRINT"▪▪   * * *  S P I R O G R A F  * * *
200 PRINT"▪▪  CANCELLO LO SCHERMO GRAFICO ? (S/N)":WAIT198,1:GETS$
210 PRINT"▪ ROTAZIONE ASSE ? (S/N)":WAIT198,1:GETA$:IFA$<>"S"THEND=π/2:GOTO230
220 INPUT" ANGOLO DI ROTAZIONE ";D:D=π/2-D*π/180
230 INPUT"▪ DENTI RUOTA FISSA ";RF%:INPUT" DENTI RUOTA MOBILE ";RM%
240 T=RF%/RM%:T1=T-1:IFT<=1THEN230
250 INPUT" DECENTRAMENTO ";DC:Z=DC*90/T:IFDC>1THEN250
260 MX=RF%:FORX=1TORF%:IFX*RF%/RM%=INT(X*RF%/RM%)THENMX=X*RF%/RM%:X=RF%
270 NEXT:MX=2*π*MX/T:R=90*(1-1/T)
280 PRINT"▪ INCREMENTO FISSO O VARIABILE ? (F/V)":WAIT198,1:GETA$
290 V=MX/400:IFA$="F"THENV=π/80
300 X1=(Z+R)*COS(D):Y1=(Z+R)*SIN(D):IFS$="S"THEN:←CLEAR
310 ←GRAF11,1:←COLOR1
320 FORA=0TOMXSTEPV:X=FNX(A):Y=FNY(A):←DRAWX,Y,0,X1,Y1,0:X1=X:Y1=Y:NEXT
330 WAIT198,1:GETA$:←TEXT6,14
340 PRINT"▪  STAMPO IL DISEGNO ? (S/N)":WAIT198,1:GETA$:IFA$="S"THEN:←GPRINT0
350 GOTO200
360 :
```

per visualizzare correttamente figure circolari. Se sul vostro schermo dovessero comparire **ovali** anzichè **circonferenze**, sarà possibile intervenire con una certa facilità modificando il valore riportato nel listato.

Il programma si presta ad una quantità inesauribile di applicazioni, dalle semplici presentazioni di propri programmi alla ricerca di più sofisticati algoritmi di elaborazione.

Con Amiga è facile inserire modifiche ed osservare subito l'effetto ottenuto: i tempi di elaborazione sono infatti molto brevi

```
130 REM  ***  VERSIONE SIMON'S BASIC  ***
140 :
150 DIM R,Z,A,T1,D,X,Y,T,MX,X1,Y1,RF%,RM%,DC,A$,S$,V,G$,P$
160 DEF FNX(X)=160+R*COS(X+D)+Z*COS(-X*T1+D)
170 DEF FNY(X)=100-(R*SIN(X+D)+Z*SIN(-X*T1+D)):P$="PREMI UN TASTO"
180 COLOUR14,6:NRM:PRINT"■■   * * *  S P I R O G R A F  * * *
190 PRINT"■■  CANCELLO LO SCHERMO GRAFICO ? (S/N)":WAIT198,1:GETS$
200 PRINT"■ ROTAZIONE ASSE ? (S/N)":WAIT198,1:GETA$:IFA$<>"S"THEND=π/2:GOTO220
210 INPUT" ANGOLO DI ROTAZIONE ";D:D=π/2-D*π/180
220 INPUT"■ DENTI RUOTA FISSA ";RF%:INPUT" DENTI RUOTA MOBILE ";RM%
230 T=RF%/RM%:T1=T-1:IFT<=1THEN220
240 INPUT" DECENTRAMENTO ";DC:Z=DC*90/T:IFDC>1THEN240
250 MX=RF%:FORX=1TORF%:IFX*RF%/RM%=INT(X*RF%/RM%)THENMX=X*RF%/RM%:X=RF%
260 NEXT:MX=2*π*MX/T:R=90*(1-1/T)
270 PRINT"■ INCREMENTO FISSO O VARIABILE ? (F/V)":WAIT198,1:GETA$
280 V=MX/400:IFA$="F"THENV=π/80
290 G$=STR$(RF%)+STR$(RM%)+STR$(DC)
300 X1=160+(Z+R)*COS(D):Y1=100-((Z+R)*SIN(D)):IFS$<>"S"THEN:CSET 2:GOTO320
310 HIRES 1,11
320 TEXT8,8,G$,1,1,8
330 FORA=0TOMXSTEPV:X=FNX(A):Y=FNY(A):LINEX1,Y1,X,Y,1:X1=X:Y1=Y:NEXT
340 TEXT200,192,P$,1,1,8
350 WAIT198,1:GETA$:NRM:TEXT200,192,P$,0,1,8
360 PRINT"■  STAMPO IL DISEGNO ? (S/N)":WAIT198,1:GETA$:IFA$="S"THEN:COPY
370 TEXT8,8,G$,0,1,8:GOTO190
380 :
```

IL C/64 MISURA LUCE E CALORE

Una manciata di componenti elettronici consente di trasformare il popolare computer in una macchina di acquisizione dati

di Maurizio Fuschetto

Abbiamo molti esempi di come i computers possano controllare la vita di noi esseri umani: avremo certamente visto come i robot lavorano e saldano pezzi meccanici, o come alcuni elaboratori (collegati ad opportuni sensori) riescano a percepire l'avvicinarsi di una persona. Con il circuito presentato in queste pagine cercheremo di ottenere qualcosa di simile.

In effetti, ciò che viene proposto non è altro che un **convertitore analogico/digitale**. Un circuito del genere non fa altro che trasformare la tensione applicata in *ingresso* (**analogica**) in una sequenza di *segnali* (**digitali**) che il computer può elaborare.

La tensione di input può provenire da diversi tipi di sensori: luminosi, termici, a raggi infrarossi...

Verrà proposto l'utilizzo di un sensore di **luce** (detto **Fotodiodo**) ed uno di **temperatura** per ottenere rispettivamente un *Luxmetro* (molto utile in fotografia) ed un *Termometro* magari per misurare la temperatura di una stanza.

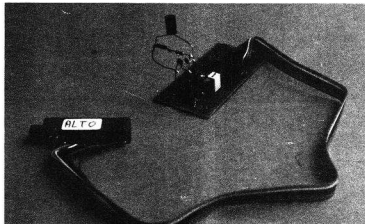
Il circuito

Il circuito è composto dai seguenti elementi, di facile reperibilità presso qualunque rivenditore di componenti elettronici:

R1: resistenza 470 ohm 1/4 W
R2: resistenza 1 Megaohm (1milione di ohm) 1/4 W
R3: resistenza 15 Kiloohm (15mila ohm) 1/4 W
R4: resistenza 10 Kiloohm (10mila ohm) 1/4 W
C1: condensatore 270 nF (nanoFarad) in poliestere
C2: condensatore 1uF (microFarad) 25V elettrolitico
C3: condensatore 47uF (microFarad) 25V elettrolitico
C4: condensatore 100nF (nanoFarad) in poliestere
C5: condensatore 100nF (nanoFarad) in poliestere
VR1: trimmer (piccola resistenza variabile) 10 Kiloohm
VR2: trimmer 50 Kiloohm
IC1: integrato CA3162
FD1: fotodiodo tipo BPW 34

Come si sarà capito, il cuore del circuito è **IC1**, un integrato convertitore A/D.

Questo, dopo aver opportunamente ela-

Il C/64, grazie alla porta utente, è in grado di acquisire dati provenienti dal "mondo" esterno

Sostituendo alcuni componenti è possibile adattare il circuito al rilevamento di misure di vario tipo

borato la tensione presente in input, restituisce i dati di output su un bus di 7 bit in modo multiplex: in pratica 4 bit formano la cifra in **modo BCD**, mentre gli altri 3 indicano se questa è la cifra delle centinaia, delle decine, o delle unità .

Il sistema BCD è molto simile al binario, ma si basa sul sistema decimale: infatti l'abbreviazione BCD vuol dire **B**inary **C**oded **D**ecimal, e cioè *Binario Codificato Decimale.*

Per rappresentare una cifra servono 4 bit, ma mentre nel sistema binario si possono ottenere 16 combinazioni con 4 bit, in BCD vengono considerate solo le prime dieci, quelle, cioè, che indicano le cifre da 0 a 9.

Riportiamo, qui di seguito, la tabella di conversione da BCD a decimale:

BCD	Decimale
0 0 0 0	= 0
0 0 0 1	= 1
0 0 1 0	= 2
0 0 1 1	= 3
0 1 0 0	= 4
0 1 0 1	= 5
0 1 1 0	= 6
0 1 1 1	= 7
1 0 0 0	= 8
1 0 0 1	= 9

Le altre combinazioni possibili non sono ritenute valide.

Per comporre, ad esempio, il numero **53** dovremo accostare i 4 bit che rappresentano 5 con i 4 bit che indicano 3:

53 = 5, **3**
0 1 0 1 , **0 0 1 1**

Il compito del computer sarà quello di verificare il bit di selezione della cifra, poi di leggere la cifra BCD e di moltiplicarla per l'opportuna potenza di dieci.

Le applicazioni

Nello schema elettrico è presente l'applicazione di un fotodiodo al convertitore, in modo da utilizzare il circuito come esposimetro (o anche Luxmetro), strumento utilissimo in fotografia, o per misurare il rendimento di una lampada per paragonarla con una di altro tipo, o per verificare se un vetro opalino è più trasparente di un altro, e per molte altre applicazioni.

In pratica il fotodiodo è un componente che genera una tensione proporzionale all'intensità di luce che lo colpisce. I componenti che lo accompagnano servono per adattare la tensione del fotodiodo a quella del convertitore.

Modificando il circuito, come indicato nello stesso schema elettrico, il convertitore diventa un termometro d'ambiente.

IC2 è un circuito integrato piccolissimo che genera una tensione proporzionale alla temperatura dello stesso, e precisamente 10 mV (milliVolt) per ogni grado centigrado. Sapendo che il massimo che il convertitore può misurare è 1V (cioè 1000 mV), potremo misurare la temperatura **da 0 a 99,9 gradi centigradi**, in quanto dovremo dividere per 10 la misura proveniente dal convertitore .

AVVERTENZE

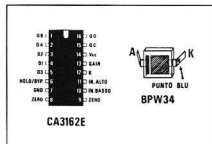
L'apparecchio descritto in queste pagine deve esser realizzato esclusivamente da chi è realmente in grado di effettuare connessioni elettriche ed elettroniche (in particolare, digitali) con la massima competenza possibile; in ogni caso è consigliabile la verifica del funzionamento prima di effettuare collegamenti con computer o suoi accessori.
La Systems Editoriale e l'autore del progetto pubblicato, pertanto, declinano ogni responsabilità da danni che dovessero eventualmente verificarsi, anche a causa di errori di stampa, di impaginazione e di progettazione hardware o software.

Il programma

Il programma, come già detto, è molto semplice e consta di 5 fasi essenziali:

1) lettura e memorizzazione della prima cifra (righe 50 / 70)

2) lettura e memorizzazione della seconda cifra (righe 80 / 100)

3) lettura e memorizzazione della terza cifra (righe 110 / 130)

Il circuito integrato è di facile reperibilità e di basso costo

4) calcolo e visualizzazione della misura effettuata (140 / 190)

5) termine (200)

Il circuito viene ovviamente collegato alla **User Port**, utilizzando le linee da PB1 a PB7, mentre PB0 resterà aperta. Segue descrizione delle connessioni alla porta utente:

pb1 - flag unità, attivo quando a livello 0
pb2 - flag decine, attivo quando a livello 0
pb3 - flag centinaia, attivo quando a livello 0
pb4 - d0 (2^0) attivo quando a livello 1
pb5 - d1 (2^1) attivo quando a livello 1
pb6 - d2 (2^2) attivo quando a livello 1
pb7 - d3 (2^3) attivo quando a livello 1

Le cifre BCD delle centinaia, delle decine e delle unità, vengono rispettivamente memorizzate nelle variabili **A**, **B** e **C**.

Siccome la vera e propria cifra BCD è presente nei 4 bit più alti del numero, viene eseguito un **And** con il numero **240** (che in esadecimale è **F0** ed in binario **1 1 1 1 0 0 0 0**); il valore ottenuto viene diviso per 16, in modo da ottenere la rotazione del numero di 4 bit verso destra.

Poi, a seconda della cifra considerata, il valore viene moltiplicato per 10 o per 100 e sommato alla variabile **M**.

Taratura

Come ogni strumento di misura, anche il convertitore ha bisogno di essere tarato, operazione comunque molto semplice. Come prima operazione bisognerà collegare l'ingresso del convertitore (pin 11 del CA3162) con il polo negativo dell'alimentazione (GND) e mandare in esecuzione il programma pubblicato. Con l'aiuto di un piccolo giravite, ruotare VR1 fino a leggere 000 sul monitor, in modo da realizzare la taratura per il minimo della scala. Per effettuare la stessa cosa per il massimo, dovremo collegare 5 resistenze da 10000 ohm tra i +5V e la massa (GND) in modo da ottenere tra il punto A e la massa una differenza di potenziale di un Volt.

Sempre con l'aiuto del giravite, ruotare VR2 fino a leggere, sul monitor, 999.

L'operazione di taratura dovrà essere effettuata **prima** di collegare qualsiasi sensore all'ingresso del convertitore.

Il circuito è ora pronto per funzionare. Potrete realizzare le applicazioni citate in apertura, magari collegando un relè sulla linea PB0 che verrà attivato al superamento di una soglia prefissata (ad esempio, la temperatura ideale della stanza, oppure l'intensità luminosa alla quale il C/64 dovrà spegnere le luci).

Un consiglio è quello di racchiudere il convertitore in un contenitore plastico o almeno di isolare il lato saldature dal tavolo, poggiando il circuito su un foglio di carta o, meglio, di plastica.

Il montaggio può essere facilmente eseguito su una basetta millefori, avendo l'accortezza di procurarsi anche il filo conduttore necessario.

```
10 PRINT"■":PRINT"            ** ACQUISIZIONE A/D **"
20 PRINTSPC(20);"BY MAUSOFT - NAPOLI"
30 FOR W=1 TO 925:NEXT W
40 POKE 56579,0:REM USER PORT POSTA IN INPUT
50 I=PEEK(56577):REM PRENDE 1'VALORE
60 V=I AND 8:IF V=8 THEN 50:REM METTE IN MEMORIA SOLO SE CIFRA CENTINAIA
70 A=I:REM ALTRIMENTI TORNA ALL'INPUT
80 I=PEEK(56577):REM PRENDE 2'VALORE
90 V=I AND 4:IF V=4 THEN 80:REM METTE IN MEMORIA SOLO SE CIFRA DECINE
100 B=I:REM ALTRIMENTI TORNA ALL'INPUT
110 I=PEEK(56577):REM PRENDE 3'VALORE
120 V=I AND 2:IF V=2 THEN 110:REM METTE IN MEMORIA SOLO SE CIFRA UNITA'
130 C=I:REM ALTRIMENTI TORNA ALL'INPUT
140 REM ** ELABORAZIONE LETTURE **
150 A=(A AND 240)/16:REM ESTRAE CIFRA BCD CENTINAIA
160 B=(B AND 240)/16:REM ESTRAE CIFRA BCD DECINE
170 C=(C AND 240)/16:REM ESTRAE CIFRA BCD UNITA'
180 M=(A*100)+(B*10)+C:REM CALCOLA MISURA
190 PRINT"■■■■■■■LA MISURA E'";M;"   ":REM VISUALIZZA
200 GET A$:IF A$="" THEN 50
210 END
```

C/64 IN ALTA RISOLUZIONE

Chi si avvicina per la prima volta alla grafica del C/64 si accorge che sono disponibili decine di Tools specifici, ma è assente la documentazione di "base" per sviluppare routines personalizzate. Il contributo (in Basic!) di un nostro lettore

di Salvatore Xompero

L'esasperante lentezza di tracciatura consente, però, di meglio comprendere la gestione della grafica

Di solito le routine grafiche di un qualsiasi Tool sono realizzate completamente in linguaggio macchina (l.m.), per ovvie questioni di velocità e quindi sono comprensibili solo da chi sa programmare, e bene, in linguaggio Assembler.

Per venire incontro ai lettori che si dilettano di solo Basic è stato deciso, quindi, di scrivere un paio di programmi, piuttosto brevi, che sono tuttavia in grado di spiegare molto bene i principi fondamentali della grafica in alta risoluzione del C/64.

La grafica

Iniziamo col vedere come il nostro amato C/64 gestisce la pagina grafica.

La pagina grafica in alta risoluzione è costituita da una matrice di punti che misura 320 per 200 (64000 punti in tutto) e che quindi occupa ben 8K di Ram. Dal momento che il chip VIC 6567, ossia il processore che si occupa della grafica, può gestire solo 16 K di memoria per volta, ossia un banco, si capisce che la pagina grafica può occupare solo due posizioni all'interno di quest'ultimo; se cosideriamo il **banco 0** (locazioni numerate da 0 a 16384), ossia quello di default all'accensione del computer, la pagina grafica può essere allocata a partire da **0** ($0) oppure a partire da **8192** ($2000).

Il C/64 ha a disposizione 4 banchi di memoria (4 * 16 K = 64K) che possono essere selezionati agendo sui due bit meno significativi della locazione **56576** ($DD00), nel seguente modo:
Poke 56576, (Peek (56576) And 252) Or n

Con **N = 0** sarà selezionato il banco **3**, con **N = 1** sarà selezionato il banco **2** e così via.

Per selezionare la posizione della **pagina** all'interno del banco bisogna invece agire sulla locazione **53272** ($D018), conosciuta anche col nome di Memory Pointer, nel seguente modo:

Poke 53272, (Peek (53272) And 240) Or N

N può essere 0, e quindi l'inizio della pagina grafica sarà uguale a quello di inizio del banco prescelto, o può essere 8 e quindi l'inizio della pagina grafica sarà dato da quello di inizio del banco sommato al numero 8192.

In questa sede faremo uso, per semplicità, del banco 0 e della pagina grafica allocata da 8192 in poi. Per attivare la pagina grafica bisogna impartire la seguente poke:

Poke 53265, Peek (53265) or 16

Ogni byte (8 bit) controlla lo stato di 8 punti, e precisamente ad ogni bit impostato a 0 corrisponde un punto spento e ad ogni bit impostato ad 1 corrisponde un punto acceso.

La corrispondenza bit - punto è alquanto complicata; ma fortunatamente viene in aiuto la matematica (aaargh!) con la seguente formula:

c = pg + 320 * int (y / 8) + 8 * int (x / 8) and 7
b = 7 -(x and 7)

I tre listati possono essere considerati come la base di partenza per implementazioni in l.m.

..dove:

PG rappresenta l'inzio della pagina grafica (8192 nel nostro caso);

X e **Y** sono le coordinate del punto;

C è la locazione dove è contenuto il punto;

B è il numero del bit relativo al punto da accendere.

Per accendere il punto di coordinate X e Y dovremo quindi effetuare la seguente operazione:

Poke C, Peek (C) or 2 ^ B

...mentre per spegnerlo dovremo digitare:

Poke C, Peek (C) And (255 * 2 ^ B)

Disegnare circonferenze

Strano ma vero, la circonferenza è la figura geometrica più facile da disegnare (a patto di conoscere almeno il seno e il coseno). Infatti come tutti sappiamo, le coordinate di un punto P qualsiasi di una circonferenza sono date dalle formule:

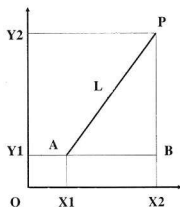
X = cos (alfa) * r
y = sin (alfa) * r

...dove R è il raggio della circonferenza in questione, mentre **alfa** è l'angolo formato dal raggio passante dal punto P con l'asse X.

Ora, sfruttando le iterazioni **For... next** possiamo far assumere all'angolo alfa tutti i valori da 0 a 2 pigreco (infatti il C/64 misura gli angoli in radianti) e plottare i relativi punti con le seguenti formule:

x = r * cos (alfa) + cx
y = r * sin (alfa) + cy

dove CX e CY sono, rispettivamente, l'ascissa e l'ordinata del centro della circonferenza.

P
Y2
L
Y1
A
B
O
X1
X2

Per ottenere una circonferenza abbastanza definita è necessario eseguire un ciclo con uno step di **0.009**.

Bisogna ricordare, inoltre, che gli angoli sono misurati in senso orario (non riesco ancora a capire perchè mamma

Commodore va sempre contro gli standard) e non in senso antiorario come siamo abituati.

Tracciare linee

Adesso che conosciamo il metodo per accendere o spegnere un punto possiamo vedere come si tracciano generiche linee.

Anche qui viene in aiuto la matematica o meglio la trigonometria (doppio aaarghh!); innanzitutto dobbiamo distinguere due tipi di tracciamento delle linee:

Linee con angolazione e lunghezza note e linee tra due punti con coordinate note.

Nel primo caso si suppone di voler tracciare una linea di determinata lunghezza, dotata di un particolare angolo di inclinazione rispetto all'asse X.

Il problema è facilmente risolvibile grazie all'uso dei cerchi.

Infatti una linea può essere immaginata come una successione di punti giacenti su circonferenze di raggio l'uno maggiore dell'altro di un valore piccolissimo.

Conoscendo l'angolo di inclinazione della retta, che chiameremo alfa, si possono calcolare le coordinate di ogni punto seguendo lo stesso procedimento usato per la circonferenza, mantenendo costante l'angolo alfa e facendo variare il raggio fino alla lunghezza voluta.

Le formule che danno le coordinate dei punti sono le stesse di quelle usate per la circonferenza.

Qualcuno si chiederà come si possa tracciare una linea conoscendo solo le coordinate dei due punti estremi; basterà calcolare la lunghezza e l'angolo di inclinazione della retta per poi plottarla con il metodo precedente.

La lunghezza della retta è data dal conosciutissimo teorema di Pitagora applicato al triangolo PAB (vedi figura):

ab = abs (x2 -x1)
pb = abs (y2 -y1)
l = sqr ((ab exp 2) + (pb exp 2))

...mentre l'angolo alfa è dato da:

alfa = Atn (y2 / x2)

Conoscendo ora i dati possiamo applicare il metodo visto sopra.

```
100 REM *** TRACCIA LINEA HI-RES ***
110 REM *** DI XOMPERO SALVATORE ***
120 REM ***        (C) 1990       ***
130 :
140 PRINT"{CLR}E' LA PRIMA VOLTA CHE GIRA IL PROGRAMMA? (S/N)"
150 GOSUB 600: IF A$="N" THEN 190
160 PRINT"{CLR}{DOWN}        ATTENDERE 16334":POKE53280,0:POKE53281,0
170 FOR I=8192 TO 16334:POKEI,0:PRINT"{HOME}"I:NEXT
180 REM *** CHIEDE COORDINATE ***
190 PRINT"{CLR}"
200 INPUT"X1 (0 - 319)";X1
210 INPUT"Y1 (0 - 199)";Y1
220 INPUT"X2";X2
230 INPUT"Y2";Y2
240 REM *** CALCOLA LUNGHEZZA LINEA **
250 REM *** E SUA ANGOLAZIONE       **
260 DX=ABS(X2-X1)
270 DY=ABS(Y2-Y1)
280 IP=SQR(DX↑2+DY↑2)
290 A=ATN(DY/DX)
300 REM ** ATTIVA LA PAGINA GRAFICA **
310 POKE53265,59:POKE53272,28:PRINT"{CLR}"
320 REM PREPARA COLORI DELLA PAGINA GRAFICA
330 FORS=1024TO2023:POKES,16:NEXTS
340 DV=0: IF X2 < X1 OR Y2 <Y1 THEN DV=1: GOTO370
350 REM *** TRACCIA LA LINEA ***
360 FOR I=0 TO IP STEP .5: GOTO 380
370 FOR I=IP TO 0 STEP -.5
380 X=I*COS(A)+X1
390 Y=I*SIN(A)+Y1
400 L=8192+(320*INT(Y/8))+(8*INT(X/8))+(YAND7)
410 B=7-(XAND7)
420 POKEL,PEEK(L)OR2↑B
430 GET A$: IF A$<>"" THEN I=IP:IF DV=1 THEN I=0
440 NEXT
450 POKE 53265,27:REM RIPRISTINA PAGINA GRAFICA
460 POKE 53272,21
470 PRINT CHR$(147)"PER RIVEDERE DIGITA:":LIST 480 - 490
480 :                     RUN 500
490 PRINT"(PREMI POI UN TASTO PER INTERROMPERE)":END
500 INPUT "COLORI (0-255)";CL
510 XX=CL/16:XA=INT(CL/16):XB=(XX-XA)*16
520 PRINT"COLORE FONDO="XA
530 PRINT"COLORE LINEA="XB
540 IF XA=XB THEN PRINT"ERRORE: COLORI EGUALI!":GOTO 500
550 GOSUB 590
560 POKE53265,59:POKE53272,28:PRINT"{CLR}"
570 FORS=1024TO2023:POKES,CL:NEXTS
580 GOSUB 600:GOTO450
590 PRINT: PRINT"(PREMI UN TASTO)"
600 GET A$:IF A$="" THEN 600
610 RETURN
620 END
```

```
90 REM  *** GRAFICA IN 3-D ***
100 FORI=8192TO16384:POKEI,0:NEXT
105 REM ** ATTIVA L'ALTA RISOLUZIONE **
107 REM ** E PULISCE LO SCHERMO      **
110 POKE53265,59:POKE53272,28:PRINT"□":POKE53280,0
115 REM *** CREA UN BUFFER   ***
117 REM *** DI 320 BYTE      ***
120 FORI=0TO319:POKE49152+I,0:POKE49700+I,199:NEXT
125 REM *** CALCOLA COORDINATE ***
130 X3=-2:FORY3=-2TO2STEP.02:GOSUB500:NEXT
140 SP=.3
150 FORX3=-2TO2STEP.02:Y3=-2
160 GOSUB500:Y3=2:GOSUB500
170 RX=X3-INT(X3/SP)*SP
200 FORY3=-2+RXTO2STEPSP
210 GOSUB500
220 NEXTY3
230 FORY3=2-RXTO-2STEP-SP
240 GOSUB500
250 NEXTY3:NEXTX3
260 X3=2:FORY3=-2TO2STEP.02:GOSUB500:NEXT
270 GETA$:IFA$=""THEN270
280 END
500 REM *** SUBROUTINE PER          ***
501 REM *** TRACCIARE IL PUNTO      ***
502 REM *** DI COORDINATE Y2 E X2   ***
505 Z3=LOG(SQR(ABS(X3↑3*Y3↑3))+.1)
510 X2=160+(Y3+X3/2)*49:Y2=100+(Z3+X3/2)*49
520 IFY2>PEEK(49152+X2)THENPOKE49152+X2,ABS(Y2):GOTO550
530 IFY2<PEEK(49700+X2)THENPOKE49700+X2,ABS(Y2):GOTO560
540 RETURN
550 IFY2<PEEK(49700+I)THENPOKE49700+X2,ABS(Y2)
560 L=8192+(320*INT((199-Y2)/8))+(8*INT(X2/8))+((199-Y2)AND7):B=7-(X2AND7)
570 POKEL,PEEK(L)OR2↑B
580 RETURN
600 REM -- COS(X3)*SIN(X3)*LOG(ABS(X3*Y3+1)) --
610 REM -- (SIN(Y3))↑2*(COS(X3)↑2):SIN(X3*Y3*4)*COS(X3*Y3/4) --
620 REM -- (COS(X3*Y3)*SIN(X3*Y3))*LOG(ABS(SIN(X3*Y3)*COS(X3*Y3)+.1)) --
630 REM -- SIN(X3*Y3*3) --
640 REM -- SIN(SQR(ABS(X3*Y3)))
650 REM -- COS(X3*Y3) --
660 REM -- SIN(X3)*COS(X3) --
670 REM -- COS(X3)*SIN(Y3) --
680 END
```

```
1000 REM TRACCIA CIRCONFERENZA
1010 REM FORI=8192TO16334:POKEI,0:NEXT
1020 :
1030 INPUT "RAGGIO";R
1040 INPUT "CENTRO X";X1
1050 INPUT "CENTRO Y";Y1
1060 POKE 53265,59: POKE53272,28
1070 V1=1024:V2=2023:SE=16:S1=7
1080 NV=8192
1090 PRINT"□":FOR I=V1 TO V2
1100 POKE I,SE:NEXT:TV=320:OT=8
1110 ZE=0:DU=2:PI=π:ZN=.009
1120 FOR I=ZE TO DU*PI STEP ZN
1130 X=COS(I)*R+X1
1140 Y=SIN(I)*R+Y1
1150 A1=TV *INT(Y/OT):A2=OT*INT(X/OT)
1160 A3=Y AND S1:B=S1-(X AND S1)
1170 L=NV +A1+A2+A3
1180 POKEL, PEEK(L) OR DU↑B
1190 NEXT
1200 POKE 53272,21:POKE53265,27
1210 GOTO1020
```

C/128, UN RASTER TUTTO PER LUI

Una routine un po' lunghetta da digitare, ma che accontenterà gli esperti l.m. dello sfortunato computer

di Angelo Garruba

L'argomento è forse un po' "duro", ma vale la pena digitare i due listati...

Il programma **Scroller 128** è un esempio di come si possano usare simultaneamente effetti **raster** e routines di **smooth scrolling**; basta inserire una qualsiasi frase per vederne di tutti i colori (e di tutte le grandezze).

Il programma offre anche una rielaborazione cromatica degli effetti di scrolling.

Il listato l.m. è realizzato con il **monitor** interno del C/128 e presenta un buon livello di ordine e modularità (è stato infatti concepito seguendo la tencnica di sviluppo Bottom-up).

Come digitare

E' necessario dapprima digitare il listato in l.m. utilizzando, come già detto, il monitor del computer.

Per fare ciò bisogna battere **Monitor** (+ Return) e, subito dopo, **M**(+ Return).

A questo punto, armandosi di santa pazienza, digitate le varie centinaia di codici macchina pubblicati in queste pagine.

Fatto ciò, salvate il segmento di memoria pazientemente digitato con...

Save "routine", 8, 1300, 15C8

Il nome può anche non essere "Routine", purchè ci si ricordi di sostituirlo nella **riga 210** del programma Basic.

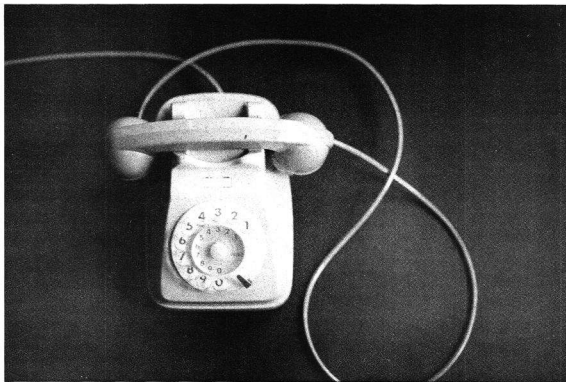
Questo, ad ogni buon conto, è scritto in modo da evitare dubbi di sorta.

Dopo aver digitato, e registrato, anche il listato Basic, provate ad impartire il solito **Run**.

Verrà chiesta una frase che, dopo alcuni secondi, apparirà di continuo, da destra a sinistra, scrollando dolcemente.

Per finire, un brutale Run / Stop + Restore.

```
110 REM SUPER SCROLLER BY
120 REM GARRUBA ANGELO
130 REM PER COMMODORE 128
140 REM
150 PRINT"E' NECESSARIO"
152 PRINT"CHE SUL DISCO"
153 PRINT"SIA PRESENTE"
154 PRINT"LA ROTINE LM"
155 PRINT"DI NOME ROUTINE"
156 PRINT
157 PRINT"RIPARTIRE CON:"
158 PRINT"RUN 170"
160 :
169 STOP
170 COLOR 4,1
180 COLOR 0,1
190 COLOR 5,15
200 SCNCLR
210 BLOAD "ROUTINE"
220 A$=""
230 INPUT "FRASE ";A$
240 LN=LEN (A$)
250 PRINT "J";
260 PRINT A$
270 FOR T=0 TO LN+10
280 A=PEEK(1024+T)
290 POKE 6282+T,A
300 NEXT
310 SCNCLR
340 :
350 FOR T=6192 TO 6233
360 POKE T,8:NEXT
370 U1=6192+8: U2=6192+42
380 FOR T=U1 TO U2 STEP 8
390 POKE T,1
400 NEXT
410 POKE 6192,11
420 :
430 FOR T=6171 TO 6190
440 READ A
450 POKE T,A
460 NEXT
470 :
480 FOR T=55896 TO 55935
490 POKE T,7
500 NEXT
510 :
520 FOR X=0 TO 9
530 READ C
540 : FOR Y=0 TO 3
550 :   FOR I = 0 TO 3
560 POKE 55696+X*4+Y*40+I,C
570 :   NEXT
580 : NEXT
590 NEXT
600 :
10000 DATA 0,9,5,6,12
10002 DATA 11,15,3,7,1
10010 DATA 1,7,3,14,11
10012 DATA 12,6,4,9,0
10020 DATA 11,12,15,15
10022 DATA 1,1,15,15,12,11
10030 SYS 5432,LN+10
10040 END
```

```
READY.

MONITOR
    PC  SR AC XR YR SP
; FB000 00 00 00 00 F8

>01300 8D 03 FF AA A9 00 A0 D0:
>01308 E0 00 F0 09 18 69 08 90:
>01310 01 C8 CA D0 F7 85 FA 84:
>01318 FB A9 04 8D 00 18 8D 01:
>01320 18 A9 00 8D 02 18 8D 03:
>01328 18 AC 02 18 B1 FA 8D 04:
>01330 18 C8 B1 FA 8D 05 18 C8:
>01338 8C 02 18 B8 50 0F 0E 04:
>01340 18 0E 04 18 0E 05 18 0E:
>01348 05 18 EA EA EA AE 04 18:
>01350 AC 05 18 20 7B 13 AE 03:
>01358 18 9D 06 18 EE 03 18 CE:
>01360 00 18 AE 00 18 D0 D7 A9:
>01368 04 8D 00 18 CE 01 18 AD:
>01370 01 18 D0 B5 A9 00 8D 00:
>01378 FF 60 EA 8A 29 C0 AA 98:
>01380 29 C0 A8 E0 00 D0 18 C0:
>01388 00 D0 03 A9 20 60 C0 80:
>01390 D0 03 A9 7B 60 C0 40 D0:
>01398 03 A9 6C 60 A9 62 60 E0:
>013A0 80 D0 18 C0 00 D0 03 A9:
>013A8 7E 60 C0 80 D0 03 A9 61:
>013B0 60 C0 40 D0 03 A9 7F 60:
>013B8 A9 FC 60 E0 40 D0 18 C0:
>013C0 00 D0 03 A9 7C 60 C0 80:
>013C8 D0 03 A9 FF 60 C0 40 D0:
>013D0 03 A9 E1 60 A9 FE 60 C0:
>013D8 00 D0 03 A9 E2 60 C0 80:
>013E0 D0 03 A9 EC 60 C0 40 D0:
>013E8 03 A9 FB 60 A9 A0 60 EA:
>013F0 EA EA EA EA EA EA EA EA:
>013F8 EA EA EA EA EA EA 24 01:
>01400 24 01 24 01 A2 00 BD 5A:
>01408 18 BC 30 18 88 D0 FD 8D:
>01410 20 D0 8D 21 D0 E8 E0 2A:
>01418 D0 EC 60 EA A0 00 AE 18:
>01420 18 E0 14 F0 0C BD 1B 18:
>01428 99 5B 18 C8 E8 E0 14 D0:
>01430 F4 A2 00 BD 1B 18 99 5B:
>01438 18 E8 C8 EC 18 18 D0 F3:
>01440 EA AE 19 18 A0 00 E0 00:
>01448 F0 0D BD 1B 18 99 6F 18:
>01450 E8 C8 CC 1A 18 D0 F3 A2:
>01458 00 BD 1B 18 99 6F 18 E8:
>01460 C8 C0 14 D0 F4 EA EE 19:
>01468 18 AD 19 18 C9 14 D0 05:
>01470 A9 00 8D 19 18 A9 14 38:
>01478 ED 19 18 8D 1A 18 EA CE:
>01480 18 18 AD 18 18 D0 05 A9:
>01488 14 8D 18 18 60 EA EA CE:
>01490 86 18 CE 86 18 AD 86 18:
>01498 C9 BF D0 64 A9 C7 8D 86:
>014A0 18 A0 00 B9 91 05 99 90:
>014A8 05 B9 B9 05 99 B8 05 B9:
>014B0 E1 05 99 E0 05 B9 09 06:
>014B8 99 08 06 C8 C0 28 D0 E3:
>014C0 EE 85 18 AD 85 18 C9 04:
>014C8 D0 18 A9 00 8D 85 18 EE:
>014D0 87 18 AC 87 18 CC 84 18:
>014D8 D0 05 A0 00 8C 87 18 B9:
>014E0 8A 18 20 00 13 AC 85 18:
>014E8 B9 06 18 8D B7 05 B9 0A:
>014F0 18 8D DF 05 B9 0E 18 8D:
>014F8 07 06 B9 12 18 8D 2F 06:
>01500 60 EA EA CE 88 18 AD 88:
>01508 18 C9 BF D0 28 A9 C7 8D:
>01510 88 18 A0 00 B9 59 06 99:
>01518 58 06 C8 C0 28 D0 F5 EE:
>01520 89 18 AC 89 18 CC 84 18:
>01528 D0 05 A0 00 8C 89 18 B9:
>01530 8A 18 8D 7F 06 60 EA EA:
>01538 78 A2 00 8E 00 FF 8D 84:
>01540 18 38 E9 01 8D 87 18 8D:
>01548 89 18 A9 03 8D 85 18 A9:
>01550 C7 8D 86 18 8D 88 18 A9:
>01558 00 8D 5A 18 8D 83 18 8D:
>01560 19 18 A9 0A 8D 18 18 A9:
>01568 14 8D 1A 18 A9 06 85 FC:
>01570 AD 12 D0 C9 52 D0 F9 20:
>01578 FE 13 C6 FC D0 04 A9 06:
>01580 85 FC AD 12 D0 C9 82 D0:
>01588 F9 AD 86 18 8D 16 D0 20:
>01590 03 15 AD 12 D0 C9 AA D0:
>01598 F9 AD 88 18 8D 16 D0 AD:
>015A0 12 D0 C9 B6 D0 F9 A9 C8:
>015A8 8D 16 D0 AD 12 D0 C9 BA:
>015B0 D0 F9 20 FE 13 A5 FC C9:
>015B8 06 D0 03 20 1C 14 20 8F:
>015C0 14 4C 70 15 EA EA 00 00:
>015C8 00 00 00 00 00 00 00 00:
```

UNA BANCA DATI PER TUTTI

Telematica, un termine ancora oscuro sebbene presente già da diversi anni nel mondo dei computer. Systems Editoriale entra a far parte del variopinto mondo della conversazione elettronica mettendo a disposizione degli utenti una banca dati aperta a tutti i lettori

di Marco Miotti

Lo scambio di dati e di programmi (purchè rigorosamente di pubblico dominio, altrimenti è vietato...) è alla base del servizio offerto dalle cosiddette *BBS*, un acronimo che significa *Bulletin Board System*, tradotto semplicemente in italiano con *Banca dati*.

E' possibile che qualche eremita del lontano Tibet non abbia ancora sentito parlare di questo particolare tipo di servizio, per cui includeremo, nel presente articolo, una breve descrizione della "filosofia" che sta alla base di una Bbs, per la gioia dei nostri lettori tibetani.

E' ormai possibile reperire facilmente, e soprattutto a basso prezzo, un modem dalle prestazioni più che dignitose, ma che, fino a qualche tempo fa, erano considerate a dir poco fantascientifiche. La velocità di 300 baud è ormai relegata in un angolino (leggi C/64 e C/128) e destinata all'utilizzo con piccoli home computer (daje!) dalle prestazioni decisamente basse (arridaje!).

Lo standard dei 1200 Baud è, infatti, quello più diffuso in Italia; il crollo dei prezzi per le macchine da 2400 Baud ha però recentemente decretato, di fatto, l'inizio della fine anche per lo standard 1200.

Un computer di modeste prestazioni è più che sufficiente per consentire, all'utente armato di modem, di affacciarsi al mondo *esterno* e di collegarsi ad un computer remoto (detto *Host*, che significa *ospite*), così da consentire una quantità di operazioni che sono limitate solo dalla fantasia dei programmatori, notoriamente alta.

BBS E MESSAGGI

Le banche dati, che offrono un servizio accessibile liberamente, consentono di abilitare nuovi utenti limitandosi, semplicemente, a richiedere una scheda anagrafica da compilarsi all'atto della prima chiamata al sistema; a discrezione dell'utente dovranno essere presentati dati più o meno veritieri, sebbene le Bbs preferiscano affidarsi alla sincerità altrui.

Una volta collegati ad una Bbs è possibile (oltre ad usufruire di altri servizi) immettere, e leggere, messaggi di ogni genere a tutto vantaggio di una sorta di *corrispondenza elettronica* che consente a diversi utenti (si presume che la Bbs ne abbia più di uno) di *colloquiare* tra loro scambiandosi informazioni utilizzando la Bbs stessa come una casella postale.

Non sono rari i casi di acquisizione di nuove amicizie... telematiche che producono messaggi del tipo "... ci vediamo Domenica a Roma" oppure "... dove vai in vacanza quest'estate?"; messaggi di chiaro stampo non-tecnico.

Lo scopo primario delle *aree messaggi* è comunque quello di consentire, a utenti diversi, la condivisione delle stesse informazioni.

Supponiamo, ad esempio, che un utente (disperato perché non riesce a far girare un determinato programma o ad installare correttamente una scheda di espansione) lanci un messaggio ad una banca dati, che offra tale servizio nella sezione, appunto, *aree messaggi*.

La richiesta di aiuto, chiaramente visibile a tutti, può essere facilmente intercettata dagli altri utenti fra i quali potrebbe trovarsi qualcuno in grado di risolvere il problema, senza contare che l'eventuale soluzione potrebbe giovare a tutti gli utenti della BBS che, curiosando tra le aree messaggi, potrebbero incontrare analoghe difficoltà in futuro.

Non mancano, ovviamente, le segnalazioni di eventuali *bugs* presenti nei programmi più diffusi ed i provvedimenti suggeriti, a tambur battente, da prendere in occasioni di epidemie virali, più o meno violente.

Tutte le Bbs possiedono almeno un'area messaggi, anche quelle private che non offrono, cioè, un servizio pubblico. Supponiamo, infatti, che la ditta *Xyz S.r.l.* decida di fornire, ai suoi agenti, la possibilità di collegarsi alla sede centrale mediante un computer portatile. La sezione messaggi, in questo caso, è fondamentale, in quanto consente, agli operatori esterni, di fornire anche in tempo reale la situazione del lavoro in corso.

Può capitare che la natura dei messaggi imponga, talvolta, una certa riservatezza, soprattutto se i diversi utenti operano in concorrenza tra loro. Anche in questo caso, però, il servizio rimane sempre affidabile dal momento che è sempre possibile ricorrere a sistemi, semplici ed efficaci, di crittografia.

FILE TRANSFER

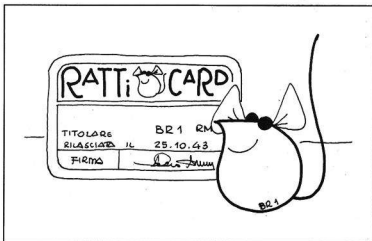
Si può affermare che il principale servizio offerto dalle Bbs era, almeno all'inizio, quello di "casella postale" elettronica.

Fino a qualche tempo fa, infatti, la velocità standard di trasmissione era di 300 Baud e qualsiasi altra operazione, che non fosse lo scambio di messaggi, era troppo costosa in termini di "scatti" telefonici; senza contare la totale inaffidabilità delle linee telefoniche commutate per la realizzazione di un sicuro file transfer.

La comparsa di accessibili protocolli di trasmissione a rilevazione e correzione automatica degli errori, nonché la già citata diffusione dei modem ad alta velocità, ha comunque contribuito al superamento dei pur gravi problemi.

E' infatti possibile realizzare trasferimenti di file alla velocità di 2400 Baud utilizzando un protocollo, quale lo *Zmodem*, che consente di trasmettere 200 Kbyte in poco più di una ventina di minuti, anche considerando la correzione di eventuali errori.

A dispetto della notoria (bassa) qualità delle linee telefoniche italiane, possiamo tranquillamente affermare che il file transfer ad alta velocità avviene nella quasi totale assenza di errori, anche sulle linee più disturbate; Zmodem prevede infatti il trasferimento di blocchi da 1024 byte che, se gli errori si susseguono frequentemente, vengono automaticamente ridotti a 512 byte. Un programma lungo 200 Kbyte è quindi costituito da 200 blocchi. In Italia, per quanto riguarda il verificarsi di errori (in ricezione o in trasmissione), questi si presentano nella quantità di circa l'1-2%, ma in questi casi vengono regolarmente "rispediti" al mittente per le correzioni da effettuare.

Un errore, di solito, si verifica solo in caso di *fischi* improvvisi sulla linea telefonica o durante la generazione di impulsi elettrici spuri.

Tutto sommato, quindi, il servizio offerto da un modem risulta abbastanza affidabile.

La **bolletta telefonica** vede chiaramente un'impennata proporzionale a tre fattori determinanti: la *durata* del collegamento, la *distanza* dall'host e l'*orario* durante il quale viene effettuato il collegamento. Con una telefonata urbana in orario di punta (circa 200 lire ogni 2 / 3 minuti) si registra un costo di circa 10000 lire per un *download* (lettura di dati) di 200000 byte, il che significa, approssimativamente, 50 lire / Kbyte.

Ovviamente lo stesso download effettuato dalla California (8000 Km di distanza) costa un pochino di più: circa 100000 lire, l'equivalente di 500 Lire / Kbyte.

Perché mai un ipotetico utente di Torino dovrebbe spendere 100 KiloLire per telefonare in Texas e "tirare giù" un programma di 200 Kbyte?

Protagonisti di questo settore sono i programmi detti di "pubblico dominio" e gli "shareware"; mentre i primi sono liberi da qualsiasi copyright e non prevedono alcun carico monetario da parte dell'utente, i programmi che ricadono nella categoria degli *shareware* sono invece subordinati ad una sorta di registrazione da parte dell'utente che, previo invio di modica cifra agli autori dei programmi, li può liberamente utilizzare.

I download effettuati riguardano esclusivamente queste categorie di programmi e, generalmente, le Bbs stesse impediscono la trattazione di file coperti da copyright... per ovvi motivi legali.

BBSYSTEMS

La *Systems Editoriale* si affaccia al mondo della telematica proponendo ai suoi lettori (e non) un servizio in perfetto stile Bbs.

Grazie, infatti, alla nuova iniziativa sarà possibile collegarsi direttamente alla redazione di *Commodore Computer Club* e di *Personal Computer* per lanciare e leggere messaggi (secondo le modalità descritte prima) e per raccogliere o inviare file in Bbs, ampliando sempre più il servizio basato principalmente sull'utente stesso.

La procedura per collegarsi con Bbsystems è la seguente:

Un utente si collega a Bbsystems e si accorge che, nell'elenco che questa propone, è assente il programma di pubblico dominio *"Pippo"*, che lui possiede.

L'utente decide, quindi, di metterlo a disposizione di Bbsystems e, dopo qualche minuto, il trasferimento è fatto; ora *"Pippo"* risiede in una particolare directory riservata agli *upload* dei lettori.

In seguito il *SysOp* (che non è altro che un addetto alla BBS), durante il suo check giornaliero si accorge della comparsa di "Pippo" nella directory degli upload e lo esamina.

Dopo aver appurato che il programma non è coperto da alcun tipo di copyright, che (nel caso si tratti di uno "shareware") non è stato modificato e che, soprattutto, non contiene virus, lo pone nella directory apposita riservata ai normali download degli utenti. Viene quindi registrato il nome dell'utente che ha offerto il software, al quale verrà inviato, a parte, anche un messaggio di ringraziamento. E', ovviamente, riservato anche un "premio" più tangibile: dopo un certo numero di Kbyte di upload verrà elevato il livello di accesso al sistema. Ciò significa che coloro che mettono software a disposizione della Bbs vedono premiata la loro "partecipazione" dal codice di accesso a directory sempre più ricche di software, di utility e di files di varia necessità.

Ed è proprio questo lo spirito delle Bbs: innescare una volontà di scambio di dati e programmi e spingere i vari utenti a cercare (o creare da soli) un numero sempre più elevato di files da mettere a disposizione della stessa Bbs.

PARAMETRI

Per connettersi con Bbsystems (operativa a pieno ritmo dal 2 Aprile 1990) è sufficiente formare il numero:

02/5249211

Il Baud rate varia, per ora, da 300 a 2400 Baud, la parità è settata come "None" con un numero di bit pari a 8 e un bit di stop.

In altre parole:

2400, N, 8, 1

Arrivederci su Bbsystems!

AREE ECHO

Sebbene Bbsystems non abbia ancora attivato questa modalità (manca ancora poco) è possibile avvalersi di speciali *Aree Echo* relative ai messaggi.

Un'area echo è definita tale quando viene condivisa da più *nodi* di una rete di Bbs, in modo che un maggior numero di utenti possano accedere agli stessi messaggi.

L'aspetto di tali aree assume i contorni di vere e proprie "conferenze" tanto che, generalmente, vengono divise per interesse (aree dedicate ai problemi riguardanti Intuition, Excel, McIntosh, gatti, rospi, astrofisica applicata e così via). La nostra Bbs non rappresenta altro che l'inizio di un discorso che vedrà orizzonti molto, molto più ampi di quello che è possibile immaginare in seguito al breve discorsetto di queste pagine. Per ora vi consigliamo di "seguirci" e, soprattutto, di procurarvi un modem, se non lo possedete. E se il computer che avete non rappresenta l'ideale per le trasmissioni telematiche?

Ma insomma: dobbiamo proprio dirvi tutto?...

POSTAMIGA

(a cura di Domenico Pavone)

Grafica e Notepad

Adoperando l'utility Notepad, non riesco a stampare su carta con caratteri diversi da quelli normali. Su video, invece, i vari Sapphire, Garnet, eccetera vengono visualizzati benissimo. Inoltre, seguendo le istruzioni del manuale, ho tentato di sostituire il driver della stampante adoperando Preferences, ma mi vengono mostrati solo due tipi: Generic e Custom. Potete darmi qualche spiegazione?

(Fabio Tiezzi - Grosseto)

Quando si adoperano particolari set di caratteri, tanto con Notepad che con altri word processor più evoluti, per ottenerne la riproduzione su carta è necessario impostare la modalità di stampa grafica, esattamente come se si inviasse alla stampante un **disegno** e non un **testo**.

In Notepad, ciò è reso possibile dall'opzione **Print As** del menu *Project*, che va impostata nella modalità **Graphic** prima di passare alla stampa vera e propria, avviata dalla scelta **Print** dello stesso menu.

Con **Draft**, invece, si otterrà sempre la riproduzione del testo nei caratteri standard adoperati dalla stampante, indipendentemente dalle caratteristiche grafiche mostrate su video. Il modo grafico, come ovvio, risulterà molto più lento in fase di stampa.

Quanto ai cosiddetti **Driver**, proprio nel caso della stampa grafica, assumono una importanza rilevante: nelle applicazioni legate al solo testo, invece, nella maggior parte dei casi è sufficiente (da Preferences) la scelta Generic per ottenere dei buoni risultati. Il fatto che non vengano visualizzati i nomi dei vari modelli di stampante, indica semplicemente... che non ci sono. Calma, calma, non è una "fregatura", solo che dalla versione 1.3 del disco di sistema sono stati trasferiti da un'altra parte, giusto per complicare l'esistenza ai nuovi utenti Amiga.

Si trovano, infatti, nel disco **Extras 1.3**, nella subdirectory **Printers** posta all'interno della directory **Devs**.

Per utilizzare quello che più si adatta alla propria stampante, è quindi necessario copiarlo nell'omonimo **Path** del disco di sistema, ovvero nella sua directory **Devs/Printers**, per poi selezionarne la scelta tramite Preferences. Copiare un file

Precisazioni per "AMIGAFACILE"

Nelle pagine del n. **75** dedicate ai comandi di **AmigaDos**, un refuso tipografico ha fatto sì che tutti i riquadri contenenti la descrizione generale degli stessi venissero titolati "*La Funzione ... (nome del comando)* ". Ad evitare equivoci sul termine **Funzione**, che può assumere un significato particolare per i programmatori, i suddetti titoli vanno intesi come "La Funzione Di...(nome del comando)", assegnando quindi a **funzione** il suo pieno significato linguistico (e non informatico) di "mansione", "compito".

Negli esempi associati al comando **Search**, è presente un **batch file** dimostrativo, le cui prime due righe sono, in realtà, da considerare come una sola (word wrap indesiderato). Il batch va quindi copiato così:

```
Search nil: Sys:c run file
If warn
echo "Manca Run!"
quit
Endif
echo "Run c'è"
```

non è impresa difficile, ma in questo caso, soprattutto se si è da poco alle prese con il Dos di Amiga, una complicazione supplementare è data proprio dal nome del disco **Extras 1.3**, che comprende uno spazio al suo interno.

Per effettuare la copia, ecco dunque la corretta sintassi nel caso si possegga un solo drive, prestando particolare attenzione alla posizione dei doppi apici...

COPY "EXTRAS 1.3:DEVS/PRINTERS/nomefile" SYS:DEVS/PRINTERS

...il tutto in una unica sequenza!

Dopo avere impartito questo comando all'interno di una finestra Shell, basterà seguire quanto richiesto direttamente dal sistema tramite i requester, preparandosi a lunghe sequele di metti-e-togli il dischetto.

Chi invece dispone di due drive, basterà che lasci il disco di sistema in **df0:**, inserisca il disco Extras nella seconda unità, e quindi impartisca...

COPY DF1:DEVS/PRINTERS/nomefile SYS:DEVS/PRINTERS

In entrambi i casi, si presuppone che si intenda copiare il file nel disco adoperato per il boot. Adoperando **Sys:**, si fa in modo che la copia venga effettuata anche se il disco di lancio non fosse il Workbench1.3, o gli fosse stato assegnato un altro nome.

Naturalmente, una volta impostato il Driver tramite le Preferences, questo andrà bene tanto per la grafica quanto per il modo testo.

Uguali, ma non identici

Ho notato uno strano comportamento di Amiga quando ha a che fare con copie di dischetti. Se, per esempio, faccio il backup di un disco con Diskcopy e poi lascio sia l'originale che la copia inseriti nei due drive, posso adoperarli (anche col Workbench) entrambi come se fossero diversi, eppure hanno lo stesso nome! Lo stesso se faccio la copia da Workbench, e poi tolgo dal nome il "Copy of...". Quando, invece, adopero un copiatore tipo Xcopy, avviene tutto il contrario: se per caso inserisco il disco-copia assieme all'originale, il computer va in Guru, oppure mi costringe a resettare. Come mai?
(Salvatore Nascimbeni - Palermo)

Una prima risposta, la si può dare a odor di logica: se due dischetti con lo stesso nome vengono riconosciuti come diversi da Amiga, vuol dire che effettivamente qualcosa di diverso lo hanno (Max Catalano dixit...).

Naturalmente potrebbero benissimo contenere dati del tutto differenti, ma proprio il caso di una riproduzione integrale ottenuta con Diskcopy (o comunque con copiatori fedeli al dos) ci pone davanti ad un assioma incontestabile: **Amiga non adopera solo il nome del disco per identificarlo** (versetto 16, parabola del Santo Kernel).

Cos'altro, allora?

Vediamolo più da vicino adoperando due diversi metodi, uno diretto (alla san Tommaso, tanto per restare in tema biblico-evangelico) ed uno, un pò più utile, in un basic piuttosto avanzato.

Per i nostri esperimenti, prima di ogni altra cosa, si esegua la copia di un floppy adoperando per esempio, da **Shell**, il comando...

Diskcopy df0: TO df1:

In alternativa, si può ricorrere alla consueta sovrapposizione di icone-disco da Workbench, e poi selezionare l'opzione Rename dal menu Workbench per eliminare dal nome della copia l'aggiunta "**Copy of...**"

Ora, per dare un'occhiata più ravvicinata ai dischetti, occorre fornirsi di un **Disk Editor**.

Può andar bene uno qualunque, ma qui ci riferiremo in particolare al **DiskX**, un programma appartenente all'area del pubblico dominio/shareware, inserito nella nostra raccolta software **Amigazzetta 7**. Lanciato l'editor, va anzitutto selezionato il drive contenente il disco da esaminare (df0:, df1:) tramite l'opzione **Unit** (per DiskX) o similare, sempre presente in questo tipo di utility: si scelga l'unità nella quale si è preventivamente inserito il floppy "origine", *quello del quale si è fatta una copia...*

Di solito, la prima visualizzazione che si ottiene è proprio quella che ci interessa, il cosiddetto **Root Block**.

Se così non fosse, si scelga, tramite l'editor, il **blocco 880, traccia 40, settore 0, Head 0**.

In pratica, l'inizio della traccia centrale del dischetto.

Un "blocco", come forse noto, è costituito da 512 byte; ebbene, se si va a "sbirciare" nella seconda metà del blocco (con DiskX = clickare su **Other Half**) scegliendo una visualizzazione dei byte in esadecimale, si potrà notare nella sezione riservata alla visualizzazione Ascii (di solito posta sulla destra dello schermo), proprio il nome del dischetto.

A questo, seguiranno un certo numero di zeri (dipende dalla lunghezza del nome), e, con DiskX nella penultima riga della sezione esadecimale, tre **long word**, ovvero tre valori espressi in 4 byte, o 32 bit che dir si voglia. Per esempio, qualcosa come...

000011B5 (dec. 4533)
000004A3 (dec. 1187)
000006A2 (1698)

Ci siamo.

Si prenda nota dei valori riscontrati (saranno diversi da quelli del nostro esempio), e si ripeta l'operazione con il disco copia.

Il nome del disco sarà lo stesso, ma i valori numerici no.

Ecco dove sta la differenza tra i due dischi.

Ma di che cosa si tratta?

Anche se la cosa non risulta immediatamente palpabile, ci troviamo di fronte alla **data di formattazione del disco**, che Amiga rappresenta in modo del tutto particolare (non penserete che 1698 corrisponda all'anno?).

La prima cifra rappresenta il numero di giorni trascorsi dal 1 gennaio 1978, il secondo il numero di minuti trascorsi dalla mezzanotte, ed il terzo i "Tick" (unità corrispondente ad 1/50 di secondo) dall'ultimo minuto.

Da questi valori, dilettandosi in una marea di calcoli, si può anche risalire ad un formato più immediatamente tangibile (giorno, mese, anno), ma la cosa per il momento non ci riguarda.

Ciò che è importante rilevare in questa sede, è come le date dei due dischi (l'originale e la copia) non possono mai risultare identiche: anche formattando un supporto magnetico e facendone immediatamente una copia, non si possono certo compiere le due operazioni nello stesso cinquantesimo di secondo...

La data, al momento della formattazione, viene prelevata da quella di sistema che, a sua volta, se non si dispone di un orologio tampone, viene impostata basandosi su quella del disco adoperato per il boot.

Anche **Diskcopy** (o **Pcopy**, e tutti i copiadischi che agiscono in modo "Dos") inserisce nel disco copia la data che Amiga considera come corrente, corretta o meno che sia.

Ovvio che alcuni copiatori "tritatutto" (chi non ne tiene una decina nel cassetto?), che tendono proprio ad evitare la sia pur minima differenza, riproducono integralmente anche i valori appena visti, generando i Guru di cui alla missiva se (per esempio) si inseriscono contemporaneamente i dischi origine e copia in una configurazione hardware dotata di due drive.

Risolto l'arcano, ecco un **altro metodo** per appurare le differenze appena viste, utile per chi non possiede un disk editor, ma soprattutto applicabile a molte altre situazioni che richiedano l'ottenimento di informazioni su un disco (e, volendo, anche sui suoi files e directory) presente in un certo drive.

Come primo passo, occorre copiare il **listato basic di queste pagine**, e salvarlo debitamente su disco.

Inoltre, prima di lanciarlo, è necessario che esso possa accedere ai due files...

Dos.bmap
Exec.bmap

...rintracciabili nella directory **BasicDemos** del disco Extras.

Per consentirlo, i meno esperti possono ricorrere ad una di queste soluzioni:

1) Rinominare il disco **Extras 1.3** in modo da eliminarne lo spazio (nuovo nome = **Extras**), e modificare le due istruzioni Library presenti nel listato in modo che diventino:

Library "Extras:basicdemos/dos.library"
Library "Extras:basicdemos/exec.library"

In questo caso, occorrerà inserire il disco Extras in uno dei drive quando, dopo il lancio del programma, verrà richiesto dal sistema.

2) Copiare (p. es.) in Ram Disk i suddetti files con una istruzione...

Copy "Extras 1.3:basicdemos/#?.bmap" Ram:

... e modificare il path dei comandi Library così:

Library "Ram:dos.library"
Library "Ram:exec.library"

3) Copiare i files .bmap nella directory o disco che si preferisce e, prima di lanciare il programma, impartire nella finestra di output un comando **Chdir** seguito dal nome del disco/directory ove i files si trovano. In questo caso, non è necessario apportare modifiche al listato.

Il programma, una volta mandato in esecuzione, è semplicissimo da usarsi: basta digitare **0** oppure **1** per esaminare il disco contenuto rispettivamente in df0: o df1:, e 3 per interromperne l'esecuzione. Dopo la scelta (se non si sono verificati errori), viene mostrato a video il nome del disco, nonché i valori che ne determinano la data di creazione.

Per assumere le informazioni sul disco, la routine adopera la funzione **Examine** della Dos Library, che può essere applicata ad un qualunque file o directory.

In pratica, una volta invocata, questa restituisce una

struttura dati contenente tutta una lunga serie di informazioni sull'elemento desiderato.

Prima, però, è necessario effettuare alcuni preparativi (si segua il listato). Anzitutto, ottenere il "Lock" (una specie di chiave di accesso) del file/directory da esaminare tramite l'omonima funzione della Dos Library.

Per i nostri scopi, forniamo come primo parametro di Lock l'indirizzo in memoria (ottenuto tramite SADD) della stringa "Df0:" oppure "Df1:" (a seconda della scelta operata prima), cui va posto obbligatoriamente un Chr$(0) in coda.

Con il secondo parametro di Lock (nel listato: access%=-2), si precisa poi che l'accesso deve essere in lettura (per scrittura, si sarebbe dovuto adoperare -1).

Ottenuto l'accesso alla directory (df0: e df1: possono in fondo considerarsi le directory principali dei dischi cui corrispondono), occorre poi riservarsi un'area di memoria dove verrà memorizzata la struttura restituita da Examine.

Operazione, questa, svolta da un'altra funzione di uso molto frequente: AllocMem, della Libreria Exec.

Per richiamarla, occorre fornire il numero di byte che si desidera allocare (nel listato = 252), ed un parametro che specifica quale tipo di memoria preferiamo: Chip, Fast, o Public.

Senza entare nei dettagli, in basic siamo costretti a fornire questa precisazione in termini strettamente numerici (in linguaggi come il C o l'Assembly ci pensano i cosiddetti Include), e 65536 ($10000) sta ad indicare memoria di tipo Public, preventivamente azzerata. Coraggio, siamo in dirittura d'arrivo.

Finalmente, infatti, si può invocare Examine, cui vanno associati come parametri il Lock prima ottenuto (variabile LL&), e l'indirizzo di inizio dell'area di memoria riservataci da Allocmem (nel listato = buffer&).

Se tutto è filato liscio, a partire dall'indirizzo Buffer& abbiamo ora la nostra brava struttura (chiamata FIB = File Info Block) "spulciabile" da basic a suon di Peek e PeekL.

In particolare, il nome del disco è presente all'indirizzo base (buffer&) più 8, per un massimo di 30 caratteri, e si concluderà con uno zero.

Le informazioni sulla "pseudo" data, sono invece contenute in tre Longword (valori a 32 bit) a partire dalla posizione 132.

Senza stare a descrivere anche i banali Print che concludono i lavori della routine, si noti come, una volta avviata, non è possibile uscirne se non selezionando l'opportuna scelta dal menu interno (Istruzione On Break...).

Il motivo di tale precauzione è legato alla necessità di "sbloccare" il Lock (con la funzione Unlock) prima di uscire dalla routine, nonché disallocare (funzione Freemem) la memoria riservata alla struttura fornita da Examine.

Studiando il listato, inoltre, si noterà la presenza di una variabile **f** che funge da Flag (segnalatore) per indicare se le varie funzioni di libreria sono state attivate almeno una volta.

In caso contrario (p. es. scegliendo di uscire dal programma senza prima averlo fatto funzionare), se si tentasse di richiamare le funzioni Unlock e Freemem in uscita, ne deriverebbe il solito, impeccabile Guru.

"Sembrava" facile, vero?

```
' Routine "Spia disco"
DECLARE FUNCTION examine% LIBRARY
DECLARE FUNCTION lock& LIBRARY
DECLARE FUNCTION allocmem& LIBRARY
LIBRARY "dos.library"
LIBRARY "exec.library": CLS: f=0
ON BREAK GOSUB nostop: BREAK ON
'
PRINT "NOME E PSEUDODATA DEL DISCO"
start:
PRINT: PRINT"0 = df0:": PRINT"1 = df1:"
PRINT "3 = fine": PRINT: PRINT"SCEGLI"
loop:
a$=INKEY$:IF a$="3" THEN
  IF f=1 GOTO fine1
  IF f=0 GOTO fine3
 END IF
IF a$ < > "0" AND a$ < > "1" THEN loop
drive$ = "df" + a$ + ":": access% = -2

LL&=lock&(SADD(drive$+CHR$(0)),access%)
IF LL&=0 THEN PRINT "ERRORE": GOTO fine3

buffer&=allocmem&(252,65536&)
IF buffer& = 0 THEN PRINT "ERRORE": GOTO fine2

leggi%=examine%(LL&,buffer&)
IF leggi%=0 THEN PRINT "ERRORE": GOTO fine1
'
f=1: nome& = buffer& + 8
FOR x=0 TO 29
 a=PEEK(nome&+x): b=PEEK(nome&+x+1)
 disk$=disk$+CHR$(a): IF b=0 THEN x=29
NEXT : COLOR 3, 0
PRINT: PRINT "DRIVE  - "; drive$
PRINT "DISCO  - "; disk$: PRINT
PRINT "giorni -"; PEEKL (buffer&+132)
PRINT "minuti -"; PEEKL (buffer&+136)
PRINT "tick   -"; PEEKL (buffer&+140)
disk$="": COLOR 1, 0: GOTO start
'
nostop:
PRINT"uscita solo da menu!!": RETURN
fine1:
CALL freemem (buffer&,252)
fine2:
CALL unlock(LL&)
fine3:
LIBRARY CLOSE: END
```

La routine in Amigabasic descritta nella risposta data ad un nostro lettore. In **neretto** *sono riportati i punti salienti, le subroutine e le variabili "importanti" del breve, ma efficace programma.*

Da Amiga a PC in italiano

Posseggo un Amiga con tastiera italiana, che utilizzo spesso per redigere testi. Dopo aver letto su Postamiga che è possibile trasferire files da un disco Amiga ad uno Ms-Dos adoperando Dos2Dos, mi sono procurato subito questo programma ed ho provato a copiare dei files di testo, che avrei così potuto adoperare in ufficio (dove si adopera un PC con dischi da 3.5). Tutto sembrava funzionare, ma, andando a leggere il testo sul PC, tutte le vocali accentate non risultano leggibili, e le righe appaiono disposte disordinatamente. Pensate sia un problema risolvibile?
(Pierluigi Capponi - Milano)

Intanto, perché si possa rendere un testo compatibile ai due "mondi", questo deve essere necessariamente redatto in formato **Ascii**, del resto "supportato" da tutti i word processor di un certo livello.

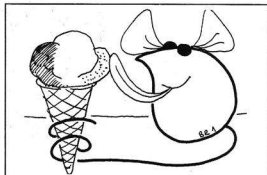
Il codice Ascii, però, anche se generalmente lo si intende come uno "standard", presenta delle differenze tra diversi computer, per lo più limitate ai codici che vanno da 128 in poi.

Il che, più terra terra, si traduce in una completa compatibilità dei normali caratteri alfanumerici, mentre quelli cosiddetti "speciali" possono differire da ambiente ad ambiente.

Tra questi caratteri speciali, sono da annoverare proprio le vocali accentate tipiche della nostra lingua.

Nel caso specifico di Amiga e Pc, esiste anche un ulteriore elemento di incompatibilità: il diverso modo di "marchiare" la fine di un paragrafo (corrispondente alla pressione di Return o Enter, per intenderci).

Amiga adopera infatti il cosiddetto Line Feed, ovvero il codice Ascii 10, mentre i Pc inseriscono un Carriage Return (ritorno carrello = Ascii 13) seguito dallo stesso codice 10 di Amiga.

Questa serie di problemi potrebbe essere certo scavalcata da opportuni programmini che "traducano" le differenze intervenendo sui files, ma c'è un modo molto più semplice ed immediato, a patto di adoperare il programma **C1-Text** (l'eccellente w/p made in Italy) per

elaborare i propri testi, o anche solo per renderli compatibili all'Ms-Dos. In pratica, basta seguire una procedura di questo tipo:

1) Dopo aver redatto (o caricato) un testo con C1-Text, prima di tutto lo si salvi come di consueto (nel formato che si preferisce: Ascii, compresso, ecc,) per averne sempre una copia di riserva in formato Amiga.

2) Si scelga dal menu **Formato File** l'opzione **Set Caratteri**, clickando nel riquadro **Ibm Pc**, e si esca dall'opzione con **Procedere**.

3) Dallo stesso menu, si selezioni ora **Parametri**,

clickando stavolta su **CR+LF** per modificare il tipo di fine linea / paragrafo in quello adatto per i Pc.

4) Salvare il file su disco scegliendo il formato Ascii.

Tutto fatto.

Non rimane che adoperare **Copy**, dopo aver attivato Dos2Dos, per trasferire il testo dal disco Amiga a quello Ms-Dos, e si potrà tranquillamente continuare ad usare le accentature italiane anche quando si ha la necessità di trasferimenti per lo più considerati incompatibili.

Come ovvio, ed a patto che il testo sia sempre in formato Ascii, la stessa procedura può anche essere invertita, per trasferire files di testo da Ms-Dos ad Amiga. Ne approfittiamo per ricordare che lavorando con Dos2Dos (soprattutto con le prime versioni) **può capitare di rovinare irrimediabilmente il dischetto Ms-Dos**, specialmente se questo presenta numerose subdirectory, o cancellature e successive ri-scritture. E' bene quindi, onde evitare spiacevoli incidenti, lavorare solo con dischetti appena formattati e contenenti, in ogni caso, copie dei files da trattare.

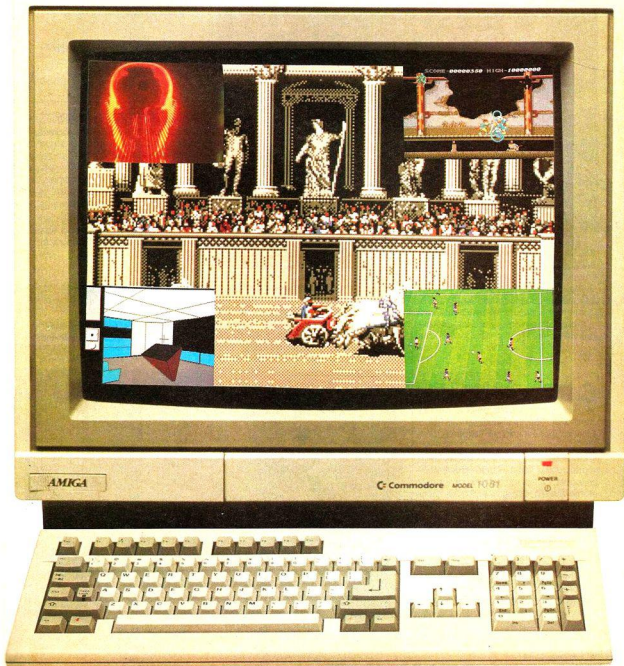
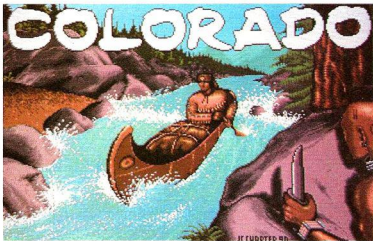
Gli originali è bene conservarli su altro dischetto.

Commodore
COMPUTER
CLUB
La rivista degli utenti di sistemi Commodore

recensioni

COLORADO

Nei panni di un Trapper vi tuffate in pericolose avventure

Computer: Amiga inespanso
Gestione: Joystick, tastiera
Tipo: Arcade adventure
Softhouse: Palace

La softhouse *Palace* gode di un buon successo soprattutto perchè ha raggruppato due famose, pur se piccole, softhouse francesi: *Delphine e Silmarils*. A loro si devono giochi famosi come **Bio Challenge** e **Future Wars**. Colorado è una classica avventura arcade controllata via icone.

Il gioco

Si vestono i panni di **David O'Brian**, un trapper (leggi: David Crockett) con l'inconfondibile cappello di pelo di marmotta(?!), che si muove in un Sud America pieno di indiani, torrenti e foreste.

Lo scopo è di trovare un testoro sulla scorta delle indicazioni date da una mappa fornita. La visione del gioco è normalmente laterale, con sprites di dimensioni maggiorate che si spostano (di poco) verso l'alto o verso il basso, procedendo da sinistra verso destra e provocando il cambiamento di scena e di fase al raggiungimento del bordo destro o sinistro.

Le armi in possesso di David sono un **tomahawk**, un **fucile** (ai tempi erano monocolpo, questo è fortunatamente a ripetizione) ed un coltello. Come prevedibile, vanno usati intelligentemente secondo l'avversario ed il tipo di combattimento. Non mancano altri oggetti da raccogliere per completare il gioco, a volta da procurarsi contrattando con qualche stregone camuffato...

Una volta esplorata completamente un'area, per spostarsi nella successiva si deve superare una fase di sapore "arcade", consistente in una corsa in **canoa**, evitando imbarcazioni indiane, rapide e scogli.

La tecnica

Il gioco si presenta molto bene, con sprites colorati, ben dimensionati e decisamente rifiniti nei particolari. Le animazioni sono molto buone. Non sono alla stessa altezza gli effetti sonori, ma comunque servono allo scopo.

Il voto

Un gioco dal sapore originale, ma privo di particolari incentivi o preziosismi tecnici. 6 1/2.

Considerato come Intro non c'è male; come gioco un po' meno

Computer: Amiga inesp. e C/64
Gestione: Joystick
Tipo: Arcade game
Softhouse: Virgin/Mastertronic

DAN DARE 3

Dan Dare è uno dei più famosi fumetti fantascientifici americani degli anni quaranta. Si tratta di una sorta di Indiana Jones dello spazio, perennemente alle prese con avventure avventurose. Non per niente questo è il terzo programma della serie: speriamo non ne facciano uno per ciascun fumetto pubblicato...

Il gioco

Dare è stato fatto prigioniero e portato su un satellite artificiale dove il cattivone di turno, **Mekon** signore dei Trens, intende portare in schiavitù l'intera razza umana, da utilizzare poi come cavia per propri esperimenti. Dan riesce a scappare e trova una navicella ma, prima di tornare sulla terra ed avvisare del pericolo, deve localizzare 50 libbre di carburante aggirandosi in uno sterminato dedalo di corridoi e camere. Inizialmente Dan è armato di una **pistola** al plasma, uno **scudo** di forza ed un **jetpac**. Il gioco si snoda quindi su vari livelli, con le consuete piattaforme, dove si devono abbattere i soliti cattivoni (compresi gli immancabili ostacoloni di fine livello), raccogliere vite extra, bombe nucleari, missili portatili et similia.

La tecnica

Le icone sono chiare ed ovvie nel significato, in contrasto con la minutezza della grafica dei personaggi. La musica è accattivante, ma gli effetti sonori sono troppo limitati per un gioco di questo tipo. La tecnica di scrolling multidirezionale è decisamente scarsa, poco fluida.

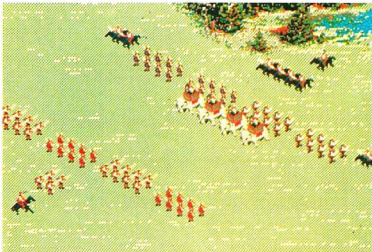
Le animazioni sono insufficienti, anche per la ridotta dimensione degli sprites: Dan, più che camminare "virilmente", sembra pattinare. Gli scenari sono abbastanza banali. La cosa riuscita meglio è l'intro.

Il voto

Un gioco mediocre, interessante per i cultori dei fumetti americani. 6-.

DEFENDER OF ROME

Una raccolta di giochi da svolgere nella Roma antica

Computer: Amiga
Gestione: Joystick
Tipo: Arcade strategico
Softhouse: Ormellysoft

L'autore del programma è nientemeno che **Kellyn Beeck**, uno degli artefici di **Defender of the Crown**: un nome, una garanzia.

Il gioco

In qualità di ufficiale al controllo di una legione della giovane antica Roma, si devono prendere le misure necessarie per difendersi dai barbari.

Contemporaneamente bisogna usare perizia politica e diplomatica per annettere nuove provincie, aumentare la propria forza e diminuire i rischi di guerre ai confini. Il motto è "pugno di ferro in guanto di velluto": o i capi delle tribù confinanti si piegano con le buone, o farete usare le cattive ai legionari.

Durante gli scontri si possono scegliere le formazioni e le tattiche di battaglia: aggiramento, aggressione a sorpresa, sfondamento eccetera. Si possono manovrare adeguatamente cavalieri, legionari ed arcieri contro le truppe nemiche per ottenere i migliori risultati a seconda delle forze in campo e dell'assetto dei nemici.

Oltre a fare la guerra, si possono costruire circhi e vestire i panni dei gladiatori, in un classico arcade di combattimento, oppure diventare piloti (non della Ferrari ma) di una biga per competere al Circo Massimo davanti agli occhi dell'Imperatore (si può anche provare a corrompere un competitore, drogare i cavalli od ingraziarsi gli dei con doni).

Insomma, non si tratta di un semplice gioco ma di una collezione omogenea di più giochi, per la maggior parte di taglio strategico, non certo di puri "tuttogrilletto".

La tecnica

La grafica è valida, disegnata a mano da veri artisti del computer. Le animazioni sono decisamente curate, come la musica.

L'interazione con il giocatore non è delle più esaltanti, ma consente di padroneggiare agevolmente il gioco.

Il voto

Un gioco consigliabile per tutti, in particolare per chi ama gli strategici con ridotte contaminazioni arcade. 9-.

La galassia è il vostro spazio vitale: governatela almeno per 1000 anni

Computer: Amiga inespanso
Gestione: Mouse, joystick
Tipo: Strategico
Softhouse: Electronic Arts

IMPERIUM

La leggendaria Electronic Arts ha assemblato un team di eccezione per produrre Imperuim. Da **Matthew Stibb**, famoso per alcuni war games, a **Nick Wilson**, apprezzato programmatore ad 8 e 16 bit. La grafica è di **Kate Cropley**, l'autrice del bellissimo **Hound of Shadow** (già recensito su C.C.C. tre mesi fa).

Il gioco

Ambientato nel 2020, si riveste la carica di Imperatore della Terra e dei pianeti alleati, e bisogna agire per rimanere al potere per almeno 1000 anni (sì, hanno scoperto il siero dell'eterna giovinezza!) oppure per restare l'unico imperatore della galassia. A scelta, si possono controllare le attività economiche, diplomatiche o militari. Abbiamo detto "a scelta" perchè si può delegare un computer molto efficiente a questi compiti.

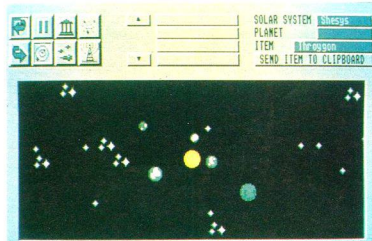
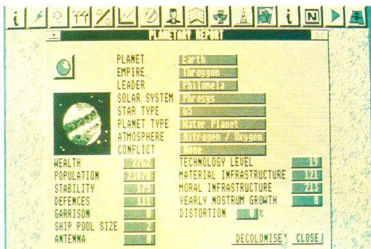
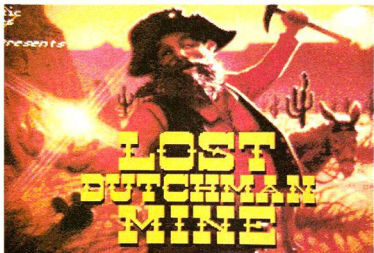
Bisogna costruire flotte di astronavi per raggiungere, invadere e controllare i pianeti confinanti. E' importante trovare anche il pianeta dei produttori di **Nostrum**, indispensabile per il filtro dell'immortalità.

Le fonti di informazioni e di controllo dell'impero sono i nostri subordinati, che devono essere incentivati con denaro, promozioni e Nostrum. Come in molti giochi analoghi, ogni 50 anni vi sono delle elezioni (anticipabili), dove si hanno chiare indicazioni dello stato della nostra carica e del grado di efficienza della politica di gestione dell'impero.

La tecnica

L'interfaccia è interamente Intuitionizzata, con un grande numero di gadget ben rifiniti. La grafica è certamente gradevole, anche se non particolarmente varia.

Il voto

Un buon programma per gli amanti del genere strategico. 7-.

DUTCHMAN MINE

Nei panni di un cercatore d'oro vi troverete in molte situazioni pericolose

Computer: Amiga
Gestione: Mouse
Tipo: Arcade di situazione
Softhouse: Magnetic Images

Una storia di muli, pionieri e metallo giallo, ambientato nel periodo in cui gli uomini erano uomini, le Colt erano Colt ed i caporali arrivavano con i nostri...

Il gioco

Lo schermo è suddiviso in due fette principali. Nel primo troviamo una grande mappa dell'area in cui si trova il nostro cercatore, che consente di spostarsi con una certa efficienza tra miniere, cittadelle, fiumi, deserto e foreste. A seconda del punto in cui ci si trova, la porzione superiore dello schermo visualizza della grafica inerente all'azione da compiere. Vi troviamo quindi ora un tavolo da poker (nel saloon), ora dei manifesti con i banditi da cercare (nell'ufficio dello Sceriffo), ora un commesso col quale contrattare l'acquisto di vettovaglie, attrezzi di lavoro, armi od animali (nei vari negozi cittadini), ora il medico con le tariffe da pagare per estrazione di pallottole, sutura di ferite da frecce, siero antiveleno di serpenti eccetera (altro che la mutua!).

Vi sono criteri da adventure/strategico nel gioco. Ad esempio, per attraversare il deserto è opportuno acquistare cibo e acqua sufficienti, nonchè una carabina per difendersi dai serpenti a sonagli.

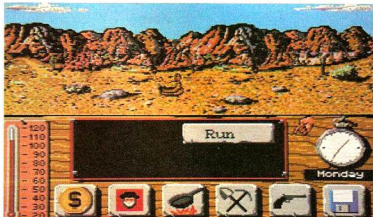
La tecnica

La grafica è carina senza essere nulla di speciale, Gli sfondi sono piuttosto rifiniti, le sceneggiature fanno uso di particolari digitalizzati, ma sono comunque di dimensioni ridotte.

La grafica delle icone è abbastanza semplificata, mentre in generale si lamenta una eccessiva riduzione delle dimensioni per la suddivisione dello schermo in due parti. La colonna sonora è tra le peggiori che ci è capitato di ascoltare: un'accozzaglia di suoni difficilmente decifrabili; appena migliori risultano gli effetti sonori "puri".

Il voto

Un programma senza molte pretese, certamente giocabile e con sufficienti incentivi, ma tecnicamente non lodevole. 7--.

NINJA SPIRIT

Sfogate la vostra cattiveria (e ingenuità...) in questo gioco spaccagrugni

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Arcade spaccagrugni
Softhouse: Activision

Un gioco dalla trama semplice, dove lo scopo della vita si riduce nel prendere a colpi di baionetta i guerrieri avversari in una scena velocissima e variegata.

Il gioco

Il primo livello di gioco vede già una serie incredibile di avversari che si materializzano dappertutto. Alcuni arrivano dall'alto, altri piombano da sotto e cercano di tagliarci le p.... ehm... trafiggerci con la spada, altri si parano davanti a muso duro. Non bisogna mai stare fermi, ma colpire sempre in corsa. Sono molto utili anche i grandi salti che possiamo compiere con un secco movimento del joystick. Procedendo nel gioco, il **ninja** può armarsi con shuriken (stelle d'acciaio appuntite), bombe, spade, corde ed altro. Troviamo anche dei ninja d'oro che, se abbattuti, danno potenza, punti o poteri magici.

Non mancano i consueti guardiani di fine livello, sorprendentemente variegati.

La tecnica

Da giocare tutto di un fiato: vi è sempre almeno un nemico che si muove esattamente alla nostra stessa velocità e che ci colpirà se ci fermiamo anche un attimo.

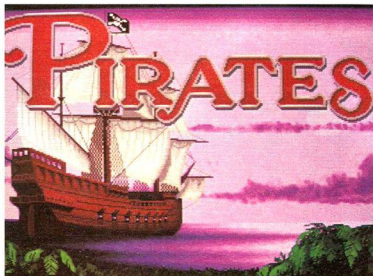
La grafica è di dimensioni ridotte (si sconsiglia di giocarle su di un TV portatile...) e molto "nervosa". Gli scenari sono piuttosto variati, alcuni molto ben disegnati.

Il voto

Consigliato agli appassionati del genere spaccagrugni, di interesse medio per gli altri. 7+.

PIRATES!

Un pirata, che "girava" tempo fa sul C/64, si aggira, ora, anche su Amiga...

Computer: Amiga inespanso
Gestione: Mouse
Tipo: Avventura animata
Softhouse: Microprose

Stranamente, si sono dovuti attendere ben due anni per vedere la versione Amiga di questo gioco, originariamente per **C/64**. Ma ne valeva la pena!

Il gioco

Si vestono i panni di un pirata (nel senso di "guerriero del mare" e non di copiatore di videogiochi...) del diciassettesimo secolo: come scenario, i Caraibi.

L'idea di base è di assumere il comando di una nave e di salpare alla ricerca di scorrerie, fama e fortuna. Si parte da un porto inglese (solitamente **Port Royale** in Giamaica) con pochi uomini e qualche soldo.

Si procede battagliando con i nemici, tenendo conto che all'inizio gli inglesi sono in guerra con tedeschi e spagnoli. Il combattimento richiama una scena con due navi viste dall'alto, in cui si deve cannoneggiare l'avversario evitando i suoi colpi e, eventualmente, procedere all'arrembaggio (cosa che avviene pressochè invariabilmente, perchè è quasi impossibile scantonare).

Solitamente lo scontro si risolve in un duello all'arma bianca, proprio con il capitano della ciurma avversaria.

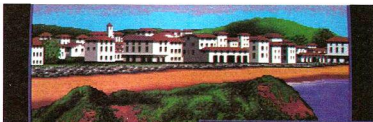
Durante il gioco si possono anche assumere altri compiti, per quattrini e/o gloria, come quello di recare messaggi da parte di qualche pezzo grosso o di liberare qualche fanciulla rapita.

La tecnica

La grafica (fissa) è molto bella, ma essenzialmente quando sono previste solo animazioni elementari. Le scene di combattimento e di animazione vera e propria non sono nulla di speciale.

Il gioco si snoda con una serie di finestre piene di opzioni predefinite (pensate a **Sinbad and the Throne of Falcon**), attraverso le quali si effettuano semplici operazioni manageriali e si decidono le azioni da compiere. Le piccole melodie che si odono ogni tanto non sono nulla di eccezionale, ma comunque risultano gradevoli.

Il voto

Buono per gli amanti del genere, scarso per la media dei videogiocatori. 7-.

PROJECTYLE

Uno strano gioco di hockey vi costringerà a smanettare con il joy per molto tempo

Computer: Amiga inespanso
Gestione: Joystick, tastiera
Tipo: Arcade sportivo
Softhouse: Electronic Arts

Un gioco sportivo futuristico sviluppato da un gruppo di programmatori ed artisti informatici noti: **Eldritch the Cat, Marc Dason** e **Steve Weatherill.**

Il gioco

Otto squadre si affrontano in una specie di hockey del futuro in un campionato di ventun settimane.

Ogni incontro viene giocato da tre squadre, con due tempi di ventidue minuti ciascuno.

Il dischetto di gioco è collocato inizialmente al centro del campo che è suddiviso in quattro zone: in tre ciascun giocatore si difende dagli attacchi provenienti dalle altre due zone, mentre nella quarta porzione di campo restante (zona mischia) si può effettuare il goal.

Il giocatore ha l'opportunità di scegliere tra le otto squadre in lizza quella che dovrà difendere la zona mischia e, tra le restanti, quelle che giocheranno nelle due zone di attacco. Le squadre hanno giocatori che possono essere addestrati per ruoli specifici nel gioco.

Per ogni partita giocata in casa ne sono previste due in trasferta, quindi possedere una squadra flessibile, in grado di adattarsi a schemi difensivi ed offensivi secondo le condizioni di gioco, è un obbligo.

La tecnica

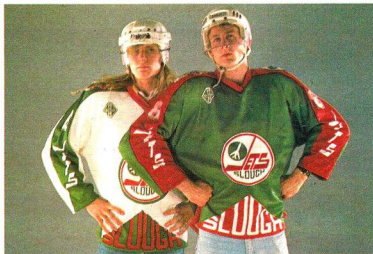
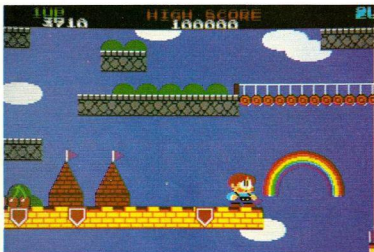
Gli sprites sono mossi molto bene, mentre gli sfondi sono ovviamente piuttosto semplificati. Gli effetti sonori sono sufficienti. Abbiamo notato qualche piccola imprecisione nel controllo degli sprites da parte del giocatore in campo, ma data la velocità dell'azione sono quasi non rilevabili.

Il voto

Un gioco originale, tecnicamente valido. 7+

RAINBOW

Un bellissimo arcobaleno riempe di colori il monitor del vostro Amiga

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Arcade a piattaforme
Softhouse: Ocean

La conversione dal coin-op della Taito è finalmente giunto nei drives di Amiga. Un gioco a piattaforme coloratissimo (rainbow significa arcobaleno), dall'inconfondibile grafica sopradimensionata ed "infantile", del tipo a scorrimento verticale sulla scia dei vari Super Mario Bros e New Zealand Story.

Il gioco

Gli eroi del gioco sono due personaggi, Bub e Bob, che arrivano direttamente da *Bubble Bobble* della Firebird, dopo avere assunto sembianze umane.

Il solito cattivone, *Boss of the Shadow* (il capo dell'ombra) ha sequestrato tutti gli abitanti dell'isola dove i nostri eroi si erano ritirati per riposare dopo la precedente avventura. Ora devono liberarli, se non altro per ripulirsi la coscienza!

Vi sono sette isole (una per colore dell'arcobaleno), ciascuna suddivisa in quattro stadi con il consueto guardiano di fine livello.

Per progredire nel giuoco bisogna salire verso l'alto sfruttando piattaforme (rammentate quelle di **Manic Miner**?) e folgorando a suon di colpi di arcobaleno i cattivoni inviati dal Boss che arrivano camminando, svolazzando, strisciando, rimbalzando eccetera.

Alcuni avversari si tasformano se colpiti dall'arcobaleno, altri muoiono assegnandoci punti, che possono essere incrementati anche raccogliendo oggetti di vario tipo (diamanti, ad esempio). Altri oggetti sono utili per proseguire nel gioco, ovvero accedere a parti altrimenti inaccessibili della scena.

Una particolarità interessante è che l'arcobaleno serve sia come arma di difesa, sia come strumento per valicare gli interspazi tra le piattaforme.

La tecnica

La grafica è di grosse dimensioni e sembra disegnata da un pittore Naif. E' uno di quei giochi che potrebbero dimostrare quanto rende avere un monitor a colori invece di un normale TV! I suoni sono semplici ed essenziali, ma gradevoli.

Il voto

Grafica ottima, suoni decenti, moltissimi incentivi, difficoltà molto progressiva. Idea semplice ma efficace. Consigliatissimo. 8 1/2.

SKY OR DIE

Gli autori sono **Rodny** e **Lester** (game Skate or Die). I protagonisti sono ancora neve e scivoloni.

Il gioco

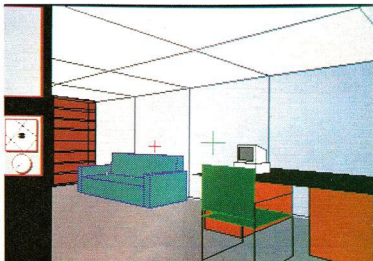
E' la solita presa in giro dei giochi invernali più seri. Cinque versioni di Hot Dog (evoluzioni sugli sci, non panini americani), affrontabili da un massimo di cinque giocatori armati di **un** joystick e **una** tastiera.

La tecnica

Per l'animazione della sola caduta i programmatori hanno disegnato circa 350 fotogrammi, immaginate quindi la nitidezza delle immagini. Gli effetti sonori sono molto buoni.

Il voto

Pochi incentivi, buona tecnica. 6 1/2.

THE COLONY

La Mindscape è specializzata in giochi strategici ed avventurosi, ma ha raggiunto poche volte la sufficienza. Questa volta propone, per Amiga inespanso e mouse, un game ispirato, per trama e tecnica, ai vari **Star Trek** e **Battlezone**.

Il gioco

Si vestono i panni del Regional Space Marshal, a bordo dell'incrociatore spaziale DAS. Improvvisamente arriva una richiesta d'aiuto da una colonia attaccata dagli alieni.

Così inizia l'avventura, che si svolge in un ambiente tridimensionale vettoriale con vista dagli occhi del giocatore, che si muove in tempo (quasi) reale nell'astronave.

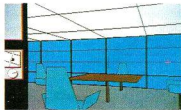
Inizialmente si deve riavviare il sistema energetico della nave, in stato di standby prima del ricevimento della missione, poi ricercare il pianeta della colonia attaccata ed ingaggiare una battaglia con gli alieni.

Il gioco consiste nel trovare i giusti elementi e le corrette parole chiave per fare funzionare l'astronave ed i sistemi di navigazione e lotta.

La tecnica

La rappresentazione si avvale di un misto di solidi poligonali e rappresentazioni puramente vettoriali (a "fili") del mondo esterno. Spostando il mouse ci si muove materialmente nello spazio e per le stanze dell'astronave. Occorre solo qualche istante per imparare a muovere il mouse con efficacia. I suoni sono quasi completamente assenti.

Il voto

Sufficiente solo per l'evidente sforzo di programmazione dell'interfaccia utente e per la grafica mista. Deludente per incentivi, trama, grafica ed effetti sonori. 6-.

TURRICAN

Un mostro, che vive in una dimensione parallela alla nostra, ci minaccia...

Computer: Amiga, C/64
Comandi: Joystick
Tipo: Shoot 'em' a piattaforma
Softhouse: Rainbow Arts

Questa softhouse tedesca si sta facendo un nome sfornando videogiochi altamente competitivi, dopo un inizio incerto ai tempi dell'Amiga 1000.

Gli autori sono **Manfred Trenz** e **Holger Schmidt**, famosi per avere convertito R-Type, chiamandolo **Denaris**, prima che uscisse la conversione ufficiale della Activision.

Ovviamente la faccenda finì a suon di avvocati e carta bollata...

Il gioco

Nel lontano passato del nostro pianeta un terribile mostro con tre teste, chiamato **Morgul**, esercitava influssi terribili sugli abitanti, rendendoli paranoici ed ossessionati da fobie di vario tipo, soprattutto di notte. Per fortuna un eroe riuscì a suonarle a Morgul ed a confinarlo in una dimensione parallela. Dopo un certo tempo, tutti si dimenticarono di lui, ma ora nuovi incubi hanno iniziato a tormentare gli abitanti del pianeta e si teme che Morgul stia per rifare capolino nella nostra dimensione. Ecco, allora, che nei panni del nuovo eroe, Turrican, si deve affrontare una serie di pericolosissime insidie per raggiungere e neutralizzare il Morgul nella dimensione parallela. Insomma, tutta 'sta storia per il solito zapper a scorrimento laterale, con un numero elevatissimo di schermate e di mostri, compresi i soliti immancabili guardiani di fine livello (più orridi e cattivi di tutti gli altri messi insieme).

La tecnica

Grafica ineccepibile: numerosi schermi, fondali curati, colori brillanti, animazione ottima degli sprites.

La softhouse dichiara che sono state inserite ben 13 fasi ambientati in cinque mondi, per una "superficie" complessiva di giuoco di circa 1200 schermi! Insomma, un'area enorme che deve essere battuta per intero per risolvere il gioco.

La musica è all'altezza della situazione, così come l'interazione col joystick.

Il voto

Un pezzo di programmazione raffinato ed ampio. Tutto da godere. 8+.

AMIGA ACTION REPLAY

Doveva esserci, e c'è. Nella sibillina affermazione può riassumersi il motivo di esistere di **Action Replay**, un supporto hardware per **Amiga 500** nato sulla scia di quanto già si era visto su computer di inferiore caratura tecnica (leggi: C/64 con le varie *Final Cartridge, Nicky*, eccetera).

Naturalmente, con le dovute proporzioni: per un supercomputer come Amiga, anche un cartuccia (termine un po' riduttivo) ad esso dedicata non poteva che allinearsi qualitativamente "in alto".

Impossibile da definire con un unico appellativo, Action Replay consente di impadronirsi totalmente di quasi tutte le risorse di Amiga, anche e soprattutto quando queste sono impegnate dall'esecuzione di un qualunque programma, (super)protetto o meno che sia.

Inutile nascondere la decisa propensione hackeristica dell'"oggetto", ma con caratteristiche tali da risultare appetibile tanto al gamofilo ammazzatutto che al più tosto degli scavamemoria tutto casa e Assembly.

Ma diamo un'occhiata da vicino. La confezione comprende, oltre la cartuccia, un manuale di trenta pagine sufficientemente chiaro ed esauriente, ed un floppy di supporto contenente alcune utility non strettamente necessarie al funzionamento del cartridge, ma indispensabili per rendere autosufficienti eventuali programmi, schermate o suoni "prelevati" da Action Reply.

L'hardware consiste in un contenitore di ridotte dimensioni, molto piatto, dal quale sporge il lungo *pettine* di collegamento da inserire nella porta di espansione sul laterale sinistro di Amiga 500. L'alloggiamento risulta pratico, con la cartuccia (peraltro molto leggera) che poggia sul piano di lavoro senza sottoporre a sforzo lo slot di espansione.

L'insieme Amiga-Cartridge forma un tutt'uno comodo ed ergonomico, tale da invitare a lasciare Action Replay collegata in permanenza, considerata anche la quasi totate non interferenza nelle normali attività del computer.

All'accensione di Amiga, come pure dopo ogni reset, il volontario "intruso"

L.179000
in vendita presso
FLOPPERIA
Viale Monte Nero, 31
20135 MILANO
tel.(02)55180484

UNA MAGICA CARTUCCIA PER GLI ASPIRANTI HACKER DI AMIGA 500

ACTION REPLAY
AMIGA
By Olaf Boehm & Joerg Zanger
(C) by Datel Electronic ltd

manifesta la sua presenza mostrando per un paio di secondi una sua spartana videata di copyright, prima di cedere lo schermo alla consueta immagine della mano prensile con dischetto workbench in dotazione.

Il game e' mio e lo gestisco io

E qui comincia il bello. Ciò che occorre fare è caricare il nostro gioco preferito. Se si tatta di un arcade mozzafiato, già qualche semplice manovra può fornire un po'di respiro a chi non è di mano ultraveloce, o non dispone di joystick megagalattici.

Dall'essenziale *piano di lavoro* (buona questa!) di Action Repaly, sporgono infatti un **miniswitch a levetta** ed una **manopolina** che (udite, udite) consentono il cosiddetto **Slow Motion**. Basta attivare l'interruttore, cui è collegato un led rosso, e, ruotando la manopola, si potrà regolare a piacimento la velocità di esecuzione di qualunque programma.

Tradotto in "gamese", questo significa che la pioggia di proiettili con la quale l'odiato nemico ci bersaglia procederà tanto lentamente quanto noi vorremo, che la pallina del "wall game" preferito non ci renderà strabici nello sforzo di seguirla mentre percorre lo schermo in frazioni di secondo, e così via smanettando.

Non male, soprattutto se si considera che si può agire sullo slow motion modificandolo in continuazione, senza dover interrompere il gioco (proverbialmente bello finchè dura).

Ma non siamo che all'inizio.

Sul frontalino di Action Replay, infatti, è presente anche un pulsante **Freeze** la cui pressione provoca un immediato ingresso in un ambiente esclusivo, dal quale si ha accesso a decine e decine di opzioni.

Il tutto, naturalmente, senza che venga persa l'esecuzione del programma che stava girando in memoria. Un comando "**X**", infatti, farà tornare Amiga al programma che stava eseguendo prima del Freeze, esattamente allo stesso punto.

Se, poi, un software più esigente non tollera la presenza del cartridge, è anche possibile escluderlo senza dovelo rimuovere fisicamente: un semplice "**XX**" dopo il Freeze, ed il gioco è fatto. In questa eventualità la cartuccia rimane inattiva anche dopo ripetuti reset, fino a che non si spegne e riaccende il computer.

L'ambiente Freeze è caratterizzato da un editor a tutto schermo, con alcune comode facilitazioni quali la possibilità di vuotare lo schermo con la pressione di **F1**, un classico "home" con **F2**, uno switch tra tastiera USA e tastiera tedesca con **F9** (no, l'italiana non c'è), nonchè la visualizzazione di tutti i comandi di Action Replay (con breve descrizione) legata al tasto **Help**.

Come se non bastasse, si dispone come ambiente di lavoro di due schermi separati, dotati di piena autonomia, in ognuno dei quali è possibile impartire i diversi comandi abilitati da Action Replay.

Molto utile, per esempio, nel caso si vogliano visualizzare e trattare due diversi schermi aperti dal game (o programma) che si stava adoperando.

Già, perchè nel caso non lo si fosse ancora capito, di un game si può gestire praticamente tutto.

```
XX  : EXIT AND KILL ACTION REPLAY: XX
Y   : DUMPMEM BINARY: Y ADR
YS  : DUMPMEM BINARY SET: YS (BYTES)
Z   : DISPLAY SYSTEM OPTIONS: Z
ZD  : SELECT DRIVE DFX: ZD 0/1
ZF  : SELECT FASTMEM: ZF 0/1
ZM  : SELECT CLEARMEM: ZM 0/1
ZV  : SELECT VIRUSTEST: ZV 0/1
?   : CALCULATE: ? VALUE (OP VALUE...)
???: SYNTAX/DATA/ADDRESS ERROR
=======================================

EDITOR-TOOLS:
=============
HELP : THIS SHORT HELP
SHIFT: NO SCROLL/PAUSE
TAB  : INSERT SPACE(S)
ESC  : ESCAPES ANY COMMAND (NOT T/TS !)
F1   : CLR + HOME
F2   : HOME
F9   : SWITCH GERMAN & USA KEYBOARD
F10  : SWITCH SCREEN (NOT IN TRAINMODE!)
LED IS OFF = READY TO EXECUTE COMMANDS!
=======================================
```

Le altre opzioni

Oltre ai programmi in toto, la nostra cartuccia dedica una notevole attenzione alla **grafica**. Quando si entra in modo Freeze, si ha infatti la possibilità di accedere allo (agli) schermi attivi per visualizzarli, salvarli, o selezionarne porzioni parziali.

La fase di visualizzazione (**P**) si rivela particolarmente importante anche perchè, dopo il rientro nell'editor, viene fornita una precisa indicazione sulle dimensioni dello schermo, eventualmente modificabili con un altro comando (**PC**) che abilita due delimitatori spostabili con il mouse.

Il tutto, poi, è salvabile su floppy nel solito formato FDos, dal quale può essere infine tradotto nel più consueto **IFF** tramite un'opzione del menu implementato dal disco Utility di Action Replay.

Analogo procedimento può essere seguito in rapporto al suono, con la possibilità di preascoltare (**H**) singolarmente ognuno dei 4 canali audio di Amiga.

Anche in questo caso, come prodotto ultimo, si può ottenere la memorizzazione in formato IFF delle caratteristiche sonore dell'audio.

Per concludere la (necessariamente) succinta rassegna delle prestazioni di Action Replay, che solo la pratica può realmente svelare in tutta la sua varietà, non resta che citare la possibilità di abilitare un modo **Trainer** per quei game dei quali non si riuscirebbe altrimenti a vedere mai i livelli più *tosti*.

Anche se non proprio semplicissimo (qualche sforzo dovrete pur farlo, prima o poi nella vita), il Trainer è favorito da alcuni comandi (**T**) che consentono, dopo l'arresto di un programma, di ricercare in memoria la locazione dove dovrebbe trovarsi memorizzato il **numero di "vite" a disposizione**.

Normalmente si avranno, in questa fase, più indirizzi (il numero 5, per esempio, può trovarsi anche in altre istruzioni macchina), per cui è necessario agire per tentativi successivi.

Dopo una prima lettura, è quindi opportuno tornare al game, perdere un'altra "vita", e Freezare ancora.

Il Trainer rimarrà attivo, e la ricerca si restringerà sempre di più fino a che il numero di vite rimaste coinciderà con un preciso indirizzo.

A questo punto, seguendo le indispensabili istruzioni del manuale, basterà modificare il valore a quell'indirizzo, e salvare l'intero programma con le nuove caratteristiche di "immortalità" (o quasi).

Chi ne avesse voglia, può addirittura personalizzare un game (o anche il semplice workbench, niente lo vieta) modificandone l'aspetto degli sprite.

Action Replay dispone infatti di uno sprite editor (comando **J**) in grado di riferirsi ad essi in base al numero o anche in base all'indirizzo di memoria nel quale sono allocate le loro descrizioni fisiche.

L'editor non è "graphic oriented", tuttavia l'uso risulta abbastanza intuitivo: lo sprite è rappresentato da una serie di righe in ognuna delle quali ogni singolo bit dell'immagine è visualizzato come l'equivalente numero colore (da 0 a 4). Per operare una modifica basta portare il cursore sui bit voluti e digitare i nuovi valori numerici.

Come più volte rimarcato, le possibili opzioni implementate da Action Replay sono davvero tante, certamente troppe per essere qui elencate, ma tutte a disposizione dei fortunati game-dipendenti possessori di un Amiga 500.

Che, tra l'altro, potranno scrollarsi di dosso ogni complesso di inferiorità nei confronti di chi detiene modelli più evoluti: Action Replay, infatti, non può al momento adattarsi a macchine come l'Amiga 2000, anche se in giro si vocifera di novità in arrivo...

Pirateria

Il fatto che sia così facile aggirare le protezioni di un programma, non esime però da un breve considerazione "etica".

Avere la copia del proprio game preferito può anche essere opportuno, scardinare le difese software di chi ha programmato il software può anche essere divertente (un "vero" hacker lo fa però a suon di linguaggio macchina), ma trarne profitto è decisamente riprovevole, oltre che illegale!

Grosso modo, si possono considerare due diversi livelli di approccio: uno riservato agli utenti più esperti, dotati di un minimo di padronanza del linguaggio macchina, ed uno accessibile a chiunque (beh, almeno la tastiera occorre saperla usare...). Per la goduria dei primi, Freeze implementa tutte le feature di un sofisticato **monitor - debugger**, persino con qualcosa in più: non mancano così i consueti **assembler** e **disassembler, fill, dump** di memoria, **find, editing** diretto dei registri, **copy** (transfer) memory, **compare, calcoli** matematici in qualunque notazione numerica, comodissima gestione dei **Break**, ed altra "robetta" assortita.

Ed ancora la possibilità di sapere tutto quello che c'è da sapere sulle correnti attività di sistema: **Librerie attive** (con indirizzo di allocazione in Ram), **task, interrupt, risorse**, e chi più ne ha più ne metta.

Sempre dedicato ai big hacker già "formati", spicca poi un comodo **Copper Assembler / Disassembler**, che consente di immettere (o leggere) espressamente le istruzioni **Wait, Skip** e **Move** senza dover elaborare in proprio le copperlist, nonchè un **editor** "bit per bit" riservato ai due CIA di Amiga.

Il facile

Se si appartiene alla più vasta categoria che, dopo quanto descritto, ha cominciato a dare i numeri (in notazione decimale, s'intende!), niente paura.

Le risorse principali di Action Replay devono ancora essere svelate, e queste sono tranquillamente accessibili a tutti, non solo alla mutazione genetica degli SpulciaRam, notoriamente dotati di otto, 16 o 32 piccoli occhi perchè possano scrutare un byte, word o long word alla volta (un occhio per bit, naturalmente).

Generalizzando un po', si può considerare il raggio di intervento diretto di Action Replay come rivolto a quattro componenti: il **programma** (inteso nella sua totalità) che gira(va) in memoria prima del Freeze, lo **schermo** o gli schermi attivi, il **suono**, e gli **sprite**.

Tutti questi elementi possono in pratica essere "estrapolati" dal loro contesto (la memoria) e salvati in formato compatto su dischi predisposti all'uopo.

Action Replay adopera per i floppy un suo particolare formato, **FDos**, per cui si rende necessario, prima di ogni altra cosa, una formattazione particolare. Nulla di più facile, naturalmente. In ambiente Freeze è disponibile un raggruppamento di comandi che consentono questo tipo di formattazione ultraveloce *(un paio di secondi!)*, come pure la selezione del drive (Df0:, Df1:, ecc.), una rapida "pulizia" della memoria, un test sulla presenza o meno di **virus**, oppure la semplice visualizzazione delle attuali condizioni di settaggio.

Su un disco così preparato, si può poi memorizzare il programma attivo al momento del Freeze con un semplice...

SA "nome programma"

Più facile di così...

Quest'ultimo può essere ricaricato in memoria in qualunque momento con un altro comando di Action Replay (**LR**), ma può anche essere affrancato dalla presenza del cartridge utilizzando una apposita utility presente nel disco a corredo.

In questo caso, quello che era il disco FDos diventerà un normale floppy AmigaDos, dotato di autoboot, che presenta al suo start un menu gestibile via mouse con i nomi dei programmi eseguibili creati (si fa per dire) con Action Replay.

Naturalmente, quando si parla di "programma", si intende la sezione che effettivamente era in esecuzione al momento del Freeze: se il software richiede l'accesso a più file di un disco, anche questi ultimi dovranno essere agibili alla copia prima effettuata. I più esperti, naturalmente, possono anche salvare l'area di memoria che interessa con i più complessi comandi Monitor.

EXECUTE

Nomefile

Nome del file batch da eseguire, eventualmente esteso con il suo percorso completo. Execute, per default, cerca il file nella directory **corrente** e nella directory **S** (che sta per **Script**) del disco di boot.
Quest'ultima, dunque, va considerata luogo di elezione per memorizzarvi file comandi, il cui nome può in questo caso essere espresso senza la specifica del percorso, quale che sia la directory corrente.
Qualora non si utilizzi Execute per l'esecuzione di un batch file preferendo ricorrere al settaggio del bit di protezione Script (Comando: **Protect nomefile +S**), l'indicazione del percorso, come pure il suo default, è da considerarsi alla stessa stregua dei programmi eseguibili.
In altre parole: adoperando il solo nome dello script (**senza** Execute), questo deve trovarsi nella directory corrente, nella directory C (non più S), oppure in una directory inserita nel Path di ricerca. Come ovvio, tanto che si adoperi Execute quanto nel caso di un batch direttamente eseguibile, è sempre possibile memorizzarlo altrove, a patto di precisarne il percorso (per esempio: Execute Ram:nomefile).

EXECUTE nomefile parametri

Parametri

Argomenti da inviare al batch file, separati tra di loro da uno spazio, ad indicare nomi di programmi, periferiche logiche e fisiche, o qualunque altra stringa accettata dai comandi Dos contenuti nel file da eseguire.
Perchè il batch file possa assumere tali argomenti, deve contenere come prima riga la direttiva **.KEY** (vedi Direttive Batch).

Nota bene

Le parole chiave di ogni comando del Dos di Amiga sono rappresentate in maiuscolo e vanno (eventualmente) adoperate così come sono. In minuscolo sono invece riprodotti i parametri che vanno ridefiniti dall'utente.

Execute

Consente l'esecuzione di un File Comandi, **Batch file** o Script file che dir si voglia.
Un batch non è altro che un normale file di testo, redatto con un qualsiasi editor che consenta una memorizzazione su disco in accordo allo standard Ascii (**Ed**, il poco brillante editor compreso nella directory **C** del disco Workbench, può anche andar bene). All'interno del file possono essere inseriti, uno per riga, tutti i comandi del Dos che si desidera vengano eseguiti in sequenza, fino a costituire una vera e propria struttura - programma.
Con la versione 1.3 del Dos è anche possibile mandare in esecuzione un batch file senza citare espressamente Execute, ma settando con **Protect** (vedi Amigafacile del n. 76) il bit **script** associato al file. In questo caso si potrà lanciare un file comandi (da Shell, **non** da Cli) invocandone il solo nome come avviene per qualunque altro programma, ma l'esecuzione sarà di fatto sempre legata ad Execute: provvederà il sistema, in nostra vece, ad utilizzarne la copia residente creata in fase di abilitazione della Shell.
Qualora si ricorra a questa feature del Dos, sarà quindi indispensabile la presenza di Execute nella lista dei comandi residenti.
In entrambi i casi, è inoltre possibile l'esecuzione in multitasking del file adoperando...
Run Execute Nomefile
...oppure...
Run Nomefile
...per la modalità diretta (= bit "script" settato con Protect).

Consigli utili

Se si pensa di adoperare spesso dei batch files, può risultare comodo **rinominare** il comando Execute in modo che ne risulti più immediata la digitazione, per esempio assegnandogli **E** come nome. In pratica, basta un comando...

Rename c:execute c:e

Disponendo del Dos 1.3, è buona norma settare tutti i batch con il bit **Script**, in modo da poterli attivare anche *(in apparenza)* senza l'ausilio di Execute. Tale procedura, tra l'altro, non esclude l'eventualità di adoperare Execute qualora (p. es.) non sia agibile la Shell, unico ambiente che consenta di sfruttare uno script in modo immediato (leggi: senza impartire execute).

Comandi Dos già pubblicati

	Sort
C.C.C. N.75	
Assign	*C.C.C. N.76*
Copy	Caratteri Speciali
Date	Delete
Dir	Format
Install	Protect
Path	Rename
Search	

Attenzione a...

Qualora si ricorresse a **Protect** per rendere direttamente eseguibile uno script file, occorre tenere presente che l'editor Ed, per suo default, memorizza il testo settandone i soli bit di protezione **RWED**. Dopo ogni eventuale correzione, occorre quindi reimpartire...

Protect Nomefile +S

...a meno che non si utilizzi espressamente Execute.

Esempi

La più semplice e frequente applicazione di un batch file, e conseguentemente di Execute, consiste nel raggruppamento di una serie di comandi del Dos per evitare di doverli digitare tutti da tastiera. Un esempio tipico:

Makedir Ram:c
Copy c:run Ram:c
Copy c:execute Ram:c
Copy c:ed ram:c
Copy :system/diskcopy Ram:c
Assign t: ram:

Supponendo di aver redatto questo banale file assegnandogli come nome **SysRam**, ogni volta che si vorrà copiare i programmi **Run, Execute, Ed** e **Diskcopy** in una directory C creata in Ram Disk, basterà impartire...

Execute SysRam

...se lo script si trova memorizzato nella directory S del disco di boot o in quella corrente. Nell'esempio, il batch provvede anche ad assegnare alla Ram Disk il device logico **T:**.

Come ovvio, possono essere aggiunte tante righe per quanti comandi si intendono copiare, provvedendo eventualmente a creare opportune directory (comando Makedir) per contenerli. In tal modo, memoria permettendo, si può in pratica trasferire in Ram (ma sarebbe molto più comodo adoperare la Rad, ovvero la Ram Disk resistente al reset) tutto il necessario per evitare ogni accesso al disco di boot.

Esigenza, questa, particolarmente sentita da chi non possiede un secondo drive.

In tal caso, però, occorrerebbe (dopo avere anche creato le relative directory in Ram Disk o Rad) accodare al batch una serie di istruzioni come...

Assign S: Ram:S
Assign C: Ram:C

...eccetera, per modificare gli assegnamenti di sistema.

L'argomento, comunque, esula dal tema di queste pagine, per cui si consiglia eventualmente di consultare gli articoli che la rivista ha dedicato (e dedica) ad una sua disamina più completa.

Per altri esempi su un uso più completo dei batch file, si veda la descrizione delle Direttive Batch e della condizione **If... Else... Endif**.

Possibili errori

Il comando Execute può anche essere inserito all'interno di un batch file, in modo da provocare l'attivazione di un file comandi esterno a quello in esecuzione. In questo caso, come pure se nel file sono incluse delle direttive batch (vedi descrizione a parte), Execute ha la necessità di creare un file intermedio nel device logico **T:**. Se questo non è stato assegnato, cerca allora di creare una directory T nel disco di sistema, che dovrebbe quindi essere abilitato alla scrittura.

In caso contrario, un requester di Amiga segnalerà la cosa e, selezionando **Cancel**, il batch abortirà la sua procedura con un *"Can't open work file..."*. Volendo evitare l'accesso al disco, da non scambiare per tentativi di aggressione virale, si può assegnare alla Ram Disk il device T: con il comando...

Assign T: Ram:

La più comune forma di errore, valida un po' per tutti i comandi, consiste in una errata (o mancante) segnalazione del percorso del file da mandare in esecuzione.

Nel caso di Execute, però, agli eventuali errori legati alla sintassi del comando vanno ad aggiungersi tutti quelli associabili ad un uso scorretto dei vari comandi inseriti nel batch file: in pratica... tutti!

DIRETTIVE BATCH

.KEY

E' la direttiva più importante, adibita alla ricezione vera e propria degli eventuali parametri associati al comando Execute. In un batch che ne fa uso, va sempre inserita come **prima riga** della sequenza di comandi, seguita dall'elenco dei nomi da sostituire ai parametri passati da Execute.

Per capirne a fondo il comportamento, prendiamo in considerazione un semplice batch file che amplia le normali funzioni del comando Copy:

```
.key sorg,dest
copy <sorg> <dest>
if exists <sorg>.info
 copy <sorg>.info <dest>.info
endif
```

Supponendo di avergli assegnato come nome **Wbcopy**, questo batch copierà, oltre al file specificato, anche l'eventuale icona ad esso associata, ovvero il file dello stesso nome ma con suffisso "**.info**".

Per ottenere quanto desiderato, andrà impartita da Shell l'istruzione...

Execute Wcopy Nome1 Nome2

...con **Nome1** che specifica il nome del file (comprensivo di un eventuale percorso) da copiare, e **Nome2** il nome (e percorso) del file destinazione.

La direttiva **.Key** (abbreviabile in **.K**), in pratica assegna a *sorg* e *dest* i parametri *Nome1* e *Nome2*, proprio come se si trattasse di due variabili, rispettando l'ordine in cui sono stati dichiarati i parametri con Execute. Vale a dire: al primo parametro (nell'esempio: Nome1) corrisponderà il primo argomento di .Key (sorg), al secondo (Nome2) l'equivalente argomento (dest), e così via.

Una volta completata l'assegnazione tramite la direttiva .Key, si potrà utilizzare in qualunque punto del batch il nome dell'argomento racchiuso tra i simboli maggiore (>) e minore (<), al quale verrà sotituito dal Dos il corrispondente parametro inviato tramite Execute.

Così, tornando al nostro esempio, prima verrà attivato un normale Copy, poi, dopo una verifica della sua esistenza (**If Exists**), viene anche copiato lo stesso file con suffisso ".info".

Quanto esposto, vale anche nella forma di esecuzione immediata di un batch (vedi **Execute** e **Protect**).

Oltre ai comuni comandi del dos, nell'ambito di un batch file possono essere adoperate alcune particolari istruzioni, tutte caratterizzate dalla presenza di un punto (.) quale primo carattere, per lo più riservate alla ricezione e sostituzione dei parametri trasmessi attraverso Execute. Genericamente, vengono definite Direttive di Execute, o Direttive Batch, e non possono essere digitate autonomamente da una riga di comando Shell o Cli.

Attenzione a...

Se un argomento specificato da **.Key** non venisse inserito quale parametro di Execute, l'esecuzione di alcuni comandi di un batch file potrebbe risultare compromessa. Per ovviare a tale inconveniente, è possibile far seguire l'argomento dall'attributo "**/a**", che ne rende obbligatoria l'immissione.

Allo stesso scopo, non volendo rendere obbligatori uno o più parametri, può essere specificata una stringa di default che sostituirà l'argomento qualora questo non fosse espresso.

La stringa andrà posta subito a ridosso del nome dell'argomento all'interno dei caratteri delimitatori **<** e **>**, preceduta dal simbolo di dollaro (**$**), in tutte le ricorrenze del parametro nell'ambito del batch file, tranne che nella riga di **.Key**.

Più facile a farsi che a dirsi, riprendiamo l'esempio del file script **Wbcopy**, che potrebbe essere così migliorato:

```
.key sorg/a,dest
copy <sorg> <dest$deflt>
if exists <sorg>.info
 copy <sorg>.info <dest$deflt>.info
endif
```

Ora, considerando ancora **Execute Wcopy Nome1 Nome2** come sintassi di esecuzione, la sequenza di comandi non verrebbe avviata se mancasse il parametro **Nome1**, cui è associato l'argomento e relativo attributo **Sorg/a**.

Nome2, invece, può essere omesso; tuttavia, senza un parametro di destinazione, il comando copy non potrebbe certo funzionare. Ecco allora che, nel caso di una sua assenza, ad ogni ricorrenza di **<dest>**, ovvero l'argomento che sarà sostituito da quanto (eventualmente) immesso in Nome2, viene fornito un default, nell'esempio **Deflt**. In definitiva: se il nome del file destinazione viene fornito quale secondo parametro di Execute, questo verrà assunto come argomento del batch, sostituendosi a tutte le ricorrenze **Dest**; in caso contrario, verrà copiato (sempre assieme alla sua eventuale icona) il file Nome1 nella directory corrente con nome Deflt. Volendo, è naturalmente possibile specificare un path diverso nel default, per esempio **<dest$Df1:deflt>**.

.BRA	.KET

Servono a sostituire, dal momento in cui queste direttive vengono dichiarate, i simboli maggiore (**.Bra**) e minore (**.Ket**) con un qualunque altro carattere.
Più chiaramente: quando il dos, durante l'esecuzione di un batch file, incontra una direttiva **.Bra X**, in tutti gli argomenti che seguono interpreterà il carattere **X** (o quello che si preferisce) come se si trattasse di un delimitatore **<**. Analogamente, con **.Ket X** verrà considerato **X** alla stregua del delimitatore finale **>**.
Per comprendere il motivo di tale implementazione, ricorriamo ad un esempio.
Si consideri il batch file, che chiameremo **Del** (in assonanza con Delete) per il compito che esso svolge: rendere più sicuro e trasparente l'uso del comando Delete.

```
.key nome/a
if exists <nome>
 ask "Cancello <nome>? (Y/N)"
if not warn
echo "Delete abortito"
quit
endif
delete <nome>
echo "File <nome> eliminato"
quit
else
echo "Il file <nome> non c'è!"
endif
```

La sua funzione è abbastanza chiara: impartendo **Execute Del Nomefile** (o solo **Del Nomefile** per il modo immediato), prima della effettiva cancellazione del file viene chiesta una conferma dell'operazione; inoltre, il batch controlla che il file da cancellare esista realmente (**if exists <nome>**), ed in caso contrario lo segnala a video.
Come già rimarcato, tutte le ricorrenze **<nome>** verranno sostituite dal parametro **Nomefile** di Execute, anche all'interno delle stringhe di **Echo** ed **Ask**, ottenendo in quest'ultimo caso la visualizzazione dell'effettivo nome del file immesso. Supponiamo, però, di voler evidenziare l'output su schermo del nome del file, inserendolo tra parentesi angolari (il solito **<** e **>**, per intenderci). Essendo questi dei simboli "riservati", vengono sempre interpretati dal dos come delimitatori di argomenti, quindi non stampabili. Ecco allora che possono tornare utili .Bra e .Ket, per esempio così:

```
.key nome/a
.bra [
.ket ]
if exists [nome]
ask "Cancello <[nome]>? (Y/N)"
if not warn
echo "Delete abortito"
quit
endif
delete [nome]
echo "File <[nome]> eliminato"
quit
else
echo "Il file <[nome]> non c'è!"
endif
```

Stavolta, grazie alle due direttive di sostituzione nelle prime righe del batch, il compito di delimitazione viene assegnato alle **parentesi quadre**, mentre quelle **angolari** diventeranno dei semplici caratteri come qualunque altro.
Ad ogni ricorrenza **[nome]** verrà dunque sostituito il nome del file da cancellare, mentre **<[nome]>** mostrerà il nome del file graficamente racchiuso tra i simboli minore (<) e maggiore (>).

.DEF

Questa direttiva consente di assegnare, una volta per tutte, un default ad un cero argomento. In pratica, svolge la stessa mansione del simbolo di dollaro, ma in maniera più lineare. Adoperando "$" (come già visto sul n. 76 di C.C.C.) è necessario specificare il default in ogni ricorrenza dell'argomento presente nel batch. Con...

.Def Argom Default

...invece, qualora il parametro **Argom** non venisse specificato con Execute, questo sarà sempre sostituito dalla stringa **Default** in qualunque punto del batch.
Rifacendoci allo stesso script file **Wbcopy** (esaminato a proposito di .Key e della specifica "dollaro"), questo potrebbe in pratica essere così strutturato:

```
.key sorg/a,dest
.def dest deflt copy <sorg>
<dest>
if exists <sorg>.info
 copy <sorg>.info <dest>.info
endif
```

Anche qui il parametro **Dest** è facoltativo, ma, qualora non immesso, verrà sostituito da **Deflt**, che diverrà il nome del file copiato nella directory corrente.

...DIRETTIVE...

.DOL	.DOT

Svolgono lo stesso compito sostitutivo esaminato a proposito di **.Bra** e **.Ket**, ma nei confronti di altri due simboli: **.Dol X** assegna ad X (che può essere qualunque altro carattere) la funzione normalmente svolta dal simbolo di dollaro ($), ovvero la rappresentazione di una stringa di default (vedi .Key).
Con **.Dot**, invece, ad essere interessato alla sostituzione è il punto. In questo caso, dopo una direttiva **.Dot X**, eventuali altre direttive andranno espresse con X come primo carattere, al posto del punto. Quindi, giusto come esempio, dopo un **.Dot !**, una successiva direttiva **.Bra** dovrà essere

IF (sintassi 1)

NOT Se specificato, inverte il significato della successiva condizione. Quindi, se p. es. *If Exists Nomefile* pone come condizione che un determinato file sia presente, con *If Not Exists Nomefile* verranno eseguite le righe successive del batch solo se quel file **non** esiste.

EXISTS Con questa parola chiave, fatta seguire dal nome di un disco, di una directory, oppure di un file eventualmente completo di percorso, ne viene testata l'esistenza (vedi esempio).

WARN ERROR FAIL L'esito della condizione If, quando viene seguita da una (ed una sola) di queste parole chiave, è posta in relazione con l'esito dell'ultimo comando eseguito.
In pratica: se non si è verificato alcun errore, il Dos imposta un codice di ritorno uguale a zero. Se, invece, l'esecuzione di un comando viene interrotta per un qualunque inconveniente, si può avere un codice di ritorno il cui valore è proporzionale alla "gravità" dell'errore.

If
Esattamente come nei linguaggi di programmazione di alto livello, consente di verificare una condizione. Se questa è soddisfatta, vengono eseguiti tutti i comandi di un batch file compresi tra la riga contenente If e la riga in cui è espresso un comando **Endif**. Qualora la condizione non risultasse valida, il flusso del batch riprenderà dalla prima riga successiva a Endif, ma può anche essere adoperata una alternativa **Else**, che attiverà le righe ad essa successive qualora la condizione di If non venisse soddisfatta. Nell'ambito di un raggruppamento di comandi condizionati da If, è inoltre possibile inserire dei salti a punti prestabiliti del batch (vedi **Skip** e **Lab**), tali da rendere "intelligente" uno script con caratteristiche simili a quelle che può assumere un programma vero e proprio.
Superfluo (o quasi) aggiungere che, perchè tutto funzioni correttamente, è indispensabile che nel device **C:** (per lo più l'omonima directory del disco di boot) siano presenti i files If, Else ed Endif.

I codici di errore

Più in dettaglio, ed in rapporto all'uso di If, potremo dunque avere:
If Not Warn - Condizione vera se non si è verificato alcun errore (codice di ritorno = 0).
If Warn - Condizione soddisfatta se il codice di errore è uguale o superiore a 5. Questo tipo di ritorno, tra l'altro, è proprio quello impostato dal comando **Ask** quando si risponde con Y (yes) al suo input.
If Error - Vero se il codice di errore è uguale o superiore a 10.
If Fail - Vero se il codice di errore è uguale o superiore a 20.
Naturalmente, in tutti i casi precedenti (non solo con warn) può essere invertita la condizione premettendo **Not** alla parola chiave.

La verifica dell'eventuale errore in cui è incorso un comando può risultare molto utile per gestirne "in proprio" le conseguenze. Esiste, infatti, una soglia di errore (per default **10**) a partire della quale un file batch viene drasticamente interrotto nella sua esecuzione. Tale soglia, però, può essere alzata ricorrendo al comando **Failat** in modo che, anche in presenza di un errore, la sequenza non si interrompa.
In tal modo, sfruttando *If* per testare il codice di ritorno, diventa possibile redirigere il flusso dello script verso una serie di istruzioni che provvedano a risolvere l'errore, o comunque ad evitarne conseguenze particolarmente dannose.

IF (sintassi 2)

Con questa forma sintattica, If consente di raffrontare due elementi.
Term1 e **Term2** rappresentano i due termini del confronto, mentre la condizione viene specificata dalle seguenti parole chiave:
EQ per indicare eguaglianza.
GT per Term1 maggiore di Term2.
GE per Term1 maggiore o uguale a Term2. Anche in questa sintassi, è possibile adoperare **Not** (p. es. Not Eq) per invertire il significato della condizione.
I parametri del raffronto, anche se numerici, vengono interpretati dal Dos come stringhe, a meno che non si aggiunga la specifica **VAL** (comunque poco utile).

Come esempio, si consideri questo breve e lineare batch file, che si limita a modificare il colore del testo digitato nell'ambito di una finestra shell:

```
.key col
if <col> eq ""
echo "1=bianco 2=nero 3=arancio"
endif
if <col> eq "1"
echo "*e[31m"
endif
if <col> eq "2"
echo "*e[32m"
endif
if <col> eq "3"
echo "*e[33m"
endif
```

Perchè svolga il suo compito, e supponendo che lo si sia chiamato **Color**, è necessario impartire **Execute Color X**, con X che può assumere un valore compreso tra 1 e 3.
Si noti come, per la condizione IF EQ sia possibile specificare due simboli di doppio apice ("") per riferirsi ad una stringa inesistente, eventualità che provoca una succinta descrizione del corretto uso del parametro **col** (o, se preferite, **X**).

La modifica del colore viene realizzata con un banale richiamo del comando **Echo**, cui viene associata una stringa di codici Ansi adatta allo scopo (argomento più volte trattato su Postamiga).

IF	term1	EQ/GT/GE	term2	VAL

Esempio

Ecco un batch file che riassume la maggior parte delle prestazioni di If, in rapporto alla sintassi fin qui esaminata:

```
.key disk
if not exists <disk>
skip abort
else
lab loop
failat 20
makedir <disk>prova
if error
ask "Disco inagibile. Riprovo? (Y/N)"
if warn
skip loop back
endif
else
delete <disk>prova
echo "Disco agibile"
endif
endif
quit
lab abort
echo "<disk> non presente!"
```

Si tratta, in pratica, della evoluzione di un batch già pubblicato sul n. 70 della rivista, atto a controllare se è possibile l'accesso ad un disco.
Utilità didattica a parte, può realmente risultare utile in molte occasioni, soprattutto se si intende rendere il più universale possibile un proprio script file.
Assegnandogli **TestDisk** come nome, il batch andrà attivato con **Execute Testdisk Nome**, dove il parametro Nome deve far riferimento al nome di un disco, o anche alla relativa unità periferica (Df0:, Df1:, eccetera.). In entrambi i casi, va obbligatoriamente digitato il simbolo due punti (:) finale.

Fulcro dello script è proprio la condizione If Error che, dopo un innalzamento della soglia di errore (**Failat 20**), intercetta l'eventuale impossibilità di aprire una directory fittizia nell'unità da esaminare *(Makedir <disk>prova)*. In questo caso, dopo aver clickato sull'opzione "Cancel" del requester che il sistema avrà esibito, il batch non si interrompe, ma si limita a segnalare la cosa ed a proporre un *input* che, volendo, da la possibilità di inserire un altro disco nel drive desiderato.
Per un uso proficuo del file, basterà sostituire alla istruzione **Echo "disco agibile"** una qualunque altra sequenza di comandi che comporti una (facoltativa) forma di accesso al disco.
Il file, per la cronaca, segnala anche se il device specificato non è presente, ricorrendo ad If Exists, coadiuvato dai comandi di "branch" Skip e Lab (si veda descrizione a parte).

SKIP e LAB

Salti condizionati (e non)

Il ricorso ad una sequenza di comandi organizzata in un batch file, si è più volte detto che consente di manipolare il Dos (quasi) alla stregua di un linguaggio di programmazione. In tale prospettiva, non può certo mancare la possibilità di **diramazione** indispensabile per qualunque linguaggio. Con AmigaDos una prestazione di questo tipo è resa possibile grazie alla condizione If, coadiuvata da una reale istruzione di salto, qualcosa di simile al **Goto** del basic.
Tale comando è **Skip**, che consente di far continuare il flusso del batch file a partire dalla posizione specificata da una "etichetta" (o "label) alfanumerica. Quest'ultima può essere inserita in qualunque punto dello script file, preceduta dalla parola chiave **Lab**.

Skip. Comando eseguibile solo nell'ambito di un batch file. Se si prova a digitarlo in modo diretto in una finestra Shell (o Cli), provvede lui stesso a ricordarcelo con un laconico messaggio: *"**Skip** must be in a command file"*.

Etichetta. Nome della Label, rappresentata da un qualunque raggruppamento di caratteri alfanumerici. La posizione verso la quale si intende dirottare il flusso delle operazioni, deve consistere in una riga contenente un'unica istruzione **Lab Etichetta**; inutile dire che il parametro Etichetta deve corrispondere alla stringa adottata per il comando Skip.
Skip può anche essere adoperato senza precisare alcun nome di label. In questo caso deve essere presente, nell'ambito del batch, una istruzione Lab anch'essa non seguita dal nome dell'etichetta.

Back. Opzionale. Il comando Skip ricerca l'etichetta verso la quale redirigere l'esecuzione solo dalla sua stessa posizione in avanti. Aggiungendo Back dopo il nome della label, il batch viene invece scandito al contrario, consentendo un salto "all'indietro" altrimenti impossibile.
Per un esempio di questa caratteristica, si veda lo script **TestDisk** proposto nella descrizione di **If**.

SKIP etichetta BACK

Esempio

Il seguente batch file...

```
.key drive/a,nome/a
ask "Formattazione veloce? (y/n)"
if warn
skip veloce
endif
format drive <drive> name <nome>
quit
lab Veloce
format drive <drive> name <nome>
quick
```

...dallo sviluppo facilmente comprensibile, applica le possibilità di salto offerte dal Dos per semplificare le operazioni di formattazione di un disco. Se lo si è redatto e memorizzato con nome **QFormat**, andrà attivato con **Execute QFormat Periferica Nomedisco**.
I parametri sono resi obbligatori dall'attributo /a presente negli argomenti di **.Key**, e specificano il drive sul quale operare (Df0:, Df1:, con i due punti finali!) ed il nome da assegnare al disco.
Il file chiederà prima se si desidera la formattazione veloce, possibile se il disco è già stato formattato almeno una volta, dopodichè attiverà il normale comando Format di AmigaDos. Il vantaggio consiste soprattutto nel non dover digitare per intero tutta la sintassi di Format, ma è sempre possibile aggiungere "feature" di propria ideazione allo script.

Quit

Interrompe l'esecuzione di un batch file.
Se questo è costituito da una semplice sequenza di comandi che non prevede particolari diramazioni o salti, non è indispensabile adoperare Quit: eseguita l'ultima istruzione del file, si avrà in ogni caso un ritorno alla finestra Dos originaria.
In molti casi, però, può essere necessario interrompere il file se si verifica una certa condizione testata da If, ed è in tali occasioni che il comando rivela la sua utilità.
Per uscire da uno script file, si può semplicemente inserire una riga contenente una istruzione **Quit**, oppure far seguire il comando da un numero che rappresenti un codice di ritorno.
Con **Quit 33**, per esempio, si avrà un ritorno alla finestra Shell (o Cli) con un messaggio *"Quit failed returncode 33"*.
L'uso del codice di ritorno è piuttosto infrequente, e può trovare applicazione solo per files di uso particolare.

QUIT

Esempio

Come esempio, ed a fini unicamente dimostrativi, ecco un breve batch che mostra le due modalità appena descritte:

```
.Key nome/a
if exists <nome>
quit
else
quit 35
endif
```

Se mandato in esecuzione con **Execute NomeBatch NomeFile**, si limita a verificare l'esistenza del file NomeFile: se ne accerta la presenza, conclude la sequenza con un semplice ritorno al Dos; in caso contrario, produrrà il già menzionato "Quit failed returncode 35".

COME COLLABORARE CON COMMODORE COMPUTER CLUB

La notevole semplicità con cui è possibile programmare un moderno computer ha permesso, ai suoi utenti, di raggiungere una formazione "professionale" di tutto rispetto.
Pochi, però, si accorgono che, appunto, alcune tecniche di programmazione vengono scoperte quasi contemporaneamente da un numero elevato di lettori della nostra rivista.
E' intuitivo che, per tali motivi, viene privilegiata la pubblicazione di articoli e programmi che possiedono spiccate caratteristiche di chiarezza e semplicità e che privilegiano l'aspetto didattico a qualsiasi altra caratteristica.
Non interessano, insomma, programmi in grado di gestire il conto in banca (ne esistono a dozzine, tutti ottimi); ma può essere utile la descrizione di un'insolita tecnica di programmazione per la creazione e la gestione velocizzata di archivi. Allo stesso modo non interessa un videogioco o una utility in l.m. se le routine sono prive del disassemblato commentato. A nessuno viene in mente di digitare una miriade di codici di cui non si conosce il significato: è molto più comodo caricare uno dei tanti games piratati. Chi intende collaborare con la rivista, quindi, rischia di perdere tempo (e denaro) se non si attiene rigorosamente alle regole già abbondantemente descritte sul n. 74 di C.C.C. ed inspiegabilmente trascurate da numerosi aspiranti collaboratori.
Pertanto, **prima** di inviare materiale, gli articoli devono **tassativamente** essere concordati ed autorizzati dal direttore in persona. E' possibile contattarlo direttamente per telefono il pomeriggio del giovedì di ogni settimana (salvo eccezioni che verranno ovviamente comunicate ai diretti interessati).
Qualunque articolo inviato (per posta o via modem) senza un preventivo accordo verrà inesorabilmente cestinato.
Nel caso si pervenisse ad un accordo, ribadiamo che sul dischetto (o nel file da nome: Label, se inviato via modem), deve essere apposta un'etichetta autoadesiva contenente le informazioni riportate in figura.
Per quento riguarda, invece, il contenuto del dischetto devono esser presenti solo ed esclusivamente i file riguardanti l'articolo.

Nome e cognome
Indirizzo (via, città, cap)
Telefono (compreso prefisso)
H/w richiesto (computer, ram minima, eventuale secondo drive, joy, mouse, ecc.)
Nome articolo concordato (e data del colloquio telefonico)
ARTICOLO.ASC
PROG.BAS
PROG.ASS
TABELLA.TAB
LETTERA.LETT

Compilazione dell'etichetta autoadesiva da apporre sul dischetto contenente l'articolo concordato. Nel file "Lettera.Lett" l'autore può trascivere eventuali note aggiuntive. Non si accettano per nessun motivo articoli e programmi su nastro.

CAMPUS
AMIGA

SOMMARIO

Nessun albero viene abbattuto per gli inserti di Commodore Computer Club, stampati su carta riciclata al 100%

UNA STRANA COPPIA DI SPRITE

Amiga è in grado di gestire sprite un po' particolari; vediamo come. Naturalmente in Assembly

di Donato De Luca

La gestione degli sprite, con Amiga, non è complessa, ma richiede uno studio attento delle strutture dei dati

Tutti i lettori avranno visto uno di quei simpatici folletti meglio conosciuti come **Sprites Hardware** poichè anche il pointer del mouse appartiene alla stessa razza.
Gli Sprites sono oggetti che possono essere posizionati e mossi indipendentemente dai **Playfields** (cioè la pagina grafica di Amiga).

Il computer possiede 8 Sprites Hardware che, alti fino a 255 pixels e larghi 16 pixels, possono assumere 4 colori.

Tuttavia il primo colore dello Sprite, o meglio (come vedremo in seguito) della **coppia** di Sprite, è il colore trasparente ed i pixels colorati con questo colore (**00** della coppia) lasceranno "vedere" quello che c'è sotto.

Gli Sprites Hardware di Amiga sono raggruppati in 4 coppie, cioè:

Sprite0 e Sprite1
Sprite2 e Sprite3
Sprite4 e Sprite5
Sprite6 e Sprite7.

Nella tabella n. 1 sono riportati i registri colore relativi ai colori delle 4 coppie di Sprites.

Oltre agli Sprites Hardware normali vi sono gli Sprites Hardware **Attached** i quali differiscono dai primi per il numero dei colori disponibili, fino a 16. Ciò è possibile in quanto uno Sprite Attached, come suggerisce anche il nome, è l'**unione** di due Sprites Hardware costituenti una coppia.

Mentre con uno Sprite normale abbiamo a disposizione 2 bit per indicare il colore di un pixel (onde i 4 colori), con uno Sprite Attached ne abbiamo a disposizione 4 (onde 16 colori),

Cop.	Col.	Nome registro	Indirizzo (esa)
0/1	0	COLOR16	$DFF1A0
	1	COLOR17	$DFF1A2
	2	COLOR18	$DFF1A4
	3	COLOR19	$DFF1A6
2/3	0	COLOR20	$DFF1A8
	1	COLOR21	$DFF1AA
	2	COLOR22	$DFF1AC
	3	COLOR23	$DFF1AE
4/5	0	COLOR24	$DFF1B0
	1	COLOR25	$DFF1B2
	2	COLOR26	$DFF1B4
	3	COLOR27	$DFF1B6
6/7	0	COLOR28	$DFF1B8
	1	COLOR29	$DFF1BA
	2	COLOR30	$DFF1BC
	3	COLOR31	$DFF1BE

Tabella n. 1: Colori delle coppie di sprites.

Col.	Nome Registro	Indirizzo
0	COLOR16	$DFF1A0
1	COLOR17	$DFF1A2
2	COLOR18	$DFF1A4
3	COLOR19	$DFF1A6
4	COLOR20	$DFF1A8
5	COLOR21	$DFF1AA
6	COLOR22	$DFF1AC
7	COLOR23	$DFF1AE
8	COLOR24	$DFF1B0
9	COLOR25	$DFF1B2
10	COLOR26	$DFF1B4
11	COLOR27	$DFF1B6
12	COLOR28	$DFF1B8
13	COLOR29	$DFF1BA
14	COLOR30	$DFF1BC
15	COLOR31	$DFF1BE

Tabella n.2: Colori Sprite Attached

ma avremo a disposizione solo 4 Sprites Attached (invece di 8), come si può anche vedere dalla tabella n.2.

DC.W $9040, $9500 ;(Word 1 , Word 2) words controllo
DC.W $0F0F, $00FF ;(Word 3 , Word 4) dati 1 riga
DC.W $0F0F, $00FF ;(Word 5 , Word 6) dati 2 riga
DC.W $0F0F, $00FF ;(Word 7 , Word 8) dati 3 riga
DC.W $0F0F, $00FF ;(Word 9 , Word 10) dati 4 riga
DC.W $0F0F, $00FF ;(Word 11 , Word 12) dati 5 riga
DC.W $0000, $0000 ;(Word 13 , Word 14) fine struttura

Tabella n.4: Struttura dei dati

Sistemi

Vi sono due sistemi per visualizzare uno Sprite: quello manuale e quello automatico.

Il Sistema Manuale, che non vedremo in questo articolo, è il più complesso, in quanto prevede che sia direttamente il 68000 ad inviare i dati riguardanti gli Sprites al loro circuito di visualizzazione.

Il sistema automatico prevede l'utilizzo di 8 canali **Dma** (detti canali Dma Sprites) per il passaggio dei dati al loro circuito generatore. L'unico lavoro che dovremo fare è allocare una struttura dati in memoria Chip, farvi puntare uno degli 8 canali Dma Sprites, e, se necessario, abilitare i canali Dma (Vedi tabella n. 3).

Esaminiamo ora una struttura dati cui far puntare un canale Dma Sprite per visualizzare uno Sprite a 4 colori di altezza 5 pixels (Tabella n.4).

Le prime due **words** sono dette di controllo, in quanto non contengono i dati della sagoma dello Sprite, ma indicano dove deve essere posizionato. La 3 e la 4 contengono i dati relativi alla prima riga dello Sprite, la 5 e la 6 quelli relativi alla seconda riga dello Sprite e così via. Infine le words 13 e 14 indicano che la struttura Sprite è finita. Infatti quando un canale Sprite

Nome	Indirizzo	Commento
SPR0PTH	$DFF120	3 bits alti
SPR0PTL	$DFF122	16 bassi
SPR1PTH	$DFF124	3 alti
SPR1PTL	$DFF126	16 bassi
SPR2PTH	$DFF128	3 alti
SPR2PTL	$DFF12A	16 bassi
SPR3PTH	$DFF12C	3 alti
SPR3PTL	$DFF12E	16 bassi
SPR4PTH	$DFF130	3 alti
SPR4PTL	$DFF132	16 bassi
SPR5PTH	$DFF134	3 alti
SPR5PTL	$DFF136	16 bassi
SPR6PTH	$DFF138	3 alti
SPR6PTL	$DFF13A	16 bassi
SPR7PTH	$DFF13C	3 alti
SPR7PTL	$DFF13E	16 bassi

Tabella n. 3: Puntatori Canali Dma Sprites

Dma trova nella struttura dati a cui sta puntando due words nulle ($0000, $0000) si ferma.

Sia sulle prime 2 words, che sulle ultime 2, ritorneremo nel corso dell'articolo in quanto necessitano di un discorso approfondito.

Prestiamo ora la nostra attenzione alle coppie words 3-4, 5-6, 7-8, 9-10, 11-12, le quali definiscono la sagoma effettiva dello Sprite. Le prime words di ogni coppia (3, 5 ,7, 9, 11) contengono i bits meno significativi della sagoma dello Sprite, mentre le altre (4, 6, 8, 10, 12) contengono quelli più significativi.

Facciamo un esempio: supponiamo di voler disegnare uno Sprite alto 5 pixels:

0 = trasparente
1 = colore 1
2 = colore 2
3 = colore 3

```
0000111122223333
0000111122223333
0000111122223333
0000111122223333
```

Il numeri che compongono l'oggetto indicano il colore da assegnare a quel particolare pixel dello Sprite.

Assegniamo al colore 1 il verde ($00f0), al colore 2 il bianco ($0fff), al colore 3 il rosso ($0f00).

Considerando il codice binario, prendiamo la prima riga dello sprite ed effettuiamo alcune sostituzioni, da decimale a binario ed incolonnando i valori (vedi tabella n.5):

0000 1111 2222 3333

COPIALO PER TELEFONO

Anche i listati presenti in queste pagine possono esser tirati giù per mezzo del modem; se, ovviamente, ne possedete uno.
La procedura per collegarsi con la nostra banca dati (attiva 24 ore su 24) è riportata su altra parte della rivista. Chi non possiede il modem (che aspettate a procurarvelo?) può tuttavia richiedere il dischetto, contenente il software, presso il nostro servizio arretrati.

La sintassi usata nel programma pubblicato è compatibile con la maggior parte degli assemblatori disponibili per Amiga

La struttura del programma consente di inserire, con una certa facilità, il controllo dello sprite per mezzo del joy

Come i più sapranno, il valore decimale nullo (0) in codice binario è dato dalla coppia **00**, 1 da **01**, 2 da **10** e 3 da **11**.

Quindi la riga dello Sprite sarà codificata, appunto, come in tabella 5 che, a sua volta, codificata in esadecimale, fornirà:

Low = $0f0f

Hight = $00ff

Ed ecco che spunta la nostra bella coppia di words:

DC.W $0F0F, $00FF

Procediamo in maniera analoga (o meglio uguale dato che le righe sono tutte uguali...) per le altre 4 righe e ci ritroviamo con la nostra

0000	1111	2222	3333	Dec./bin
0000	1111	0000	1111	Low
0000	0000	1111	1111	Hight

Tabella n. 5: Codificazione dei colori

struttura Sprite. Per gli Sprite Attached il metodo è analogo; solo che, dato che ogni pixel sarà definito da 4 bits, il procedimento sarà più lungo e noioso, (il doppio, per l'appunto) in quanto per ogni riga dello Sprite avremo bisogno di 4 words, una per ogni piano bits.

E' bene ricordare che esistono numerosi programmi di pubblico dominio (e non) che consentono di editare uno Sprite, fornendo la sua struttura Dma.

❑❒

Posizionamento sullo schermo

Prima avevamo accennato a 2 words , dette words di controllo. Vediamo ora di farne una conoscenza più approfondita.

Bits 15-8 Prima word:

Sono i bits meno significativi della posizione di start verticale (Vstart)

Bits 7-0 Prima word:

Sono i bits più significativi della posizione di start orizzontale (Hstart)

Bits 15-8 Sec. word:

Sono i bits meno significativi della posizione di stop verticale (Vstop)

Bits 7-0 Sec. word:

"Miscellanea" (Control Bits); a loro volta identificabili con:

Bit 7: Indica che quella coppia di Sprites deve essere considerata come uno Sprite Attached

Bits 6, 5, 4, 3: Non assegnati. Tuttavia l'Hardware Reference Manual consiglia (o meglio ordina..) di tenerli sempre a 0 per la compatibilità verso l'alto (...il paradiso?)

Bit 2 : E' il bit più significativo della coordinata di start verticale (Vstart)

Bit 1: E' il bit più significativo della coordinata di stop verticale (Vstop)

Bit 0: E' il bit meno significativo della coordinata di start orizzontale (Hstart)

Le posizioni di Start e Stop verticali (Vstart) possono assumere un qualsiasi valore compreso tra 0 e 262 mentre la posizione di start orizzontale (Hstart) può variare da 0 a 447. E' importante notare che Vstop dovrà essere sempre maggiore di VStart.

Più precisamente, Vstop = Vstart + N, dove N rappresenta l'altezza dello Sprite in pixels. Non rispettando tale relazione, lo Sprite non sarà visualizzato sullo schermo.

La visualizzazione di uno Sprite dipende indirettamente dalla finestra video (**Display Window**).

Per visualizzare uno Sprite nelle coordinate X=0, Y=100 (in una finestra video di coordinate di partenza X=64 e Y=44), bisogna aggiungere ad X e Y (cioè 0, 100) due valori che chiameremo **Delta X** e **Delta Y**, coordinate di Start orizzontali e verticali della finestra video.

Normalmente una finestra video ha, come coordinate di start, X = 64 e Y = 44; ne deduciamo che delta X = 64 e delta y = 44.

Esempio: con una finestra video standard, il valore per X (Hstart) sarà 0 + 64, cioè 64 ($40) mentre per Y (Vstart) sarà 100+44 cioè 144 ($90). Vstop sarà dato da Vstart + N dove N in questo caso vale 5. Vstop sarà quindi 149 ($95).

```
DC.W $9040, $9500
DC.W $0F0F, $00FF
DC.W $0F0F, $00FF
DC.W $0F0F, $00FF
DC.W $0F0F, $00FF
DC.W $0F0F, $00FF
DC.W $0000, $0000
```

Tabella n. 6: Struttura dati per sprite

Nella tabella 6 è riportata la struttura dati cui far puntare un canale Dma Sprite per visualizzare uno Sprite, alto 5 pixels, nelle coordinate X=0, Y=100 .

❑❒

Il movimento degli sprites

Uno Sprite generato con il sistema automatico può essere mosso semplicemente modificando la sua Vstart (e quindi la sua Vstop) e la sua Hstart.

Ogni volta che l'Amiga prepara l'Output Video, i dati riguardanti gli Sprites vengono passati dai canali Dma Sprite al circuito degli Sprites, il quale provvede a visualizzarli.

Se cambiamo la posizione di uno Sprite (Vstart e Vstop o/e Hstart, cioè il contenuto delle prime 2 words di controllo) prima che esso sia di nuovo visualizzato, esso, quando sarà nuovamente visualizzato apparirà in una posizione differente dalla precedente e quindi sembrerà essersi mosso.

Tuttavia si deve stare molto attenti nel compiere questa operazione: infatti se modifichiamo il contenuto delle 2 words di controllo (cioè Vstart, Vstop, Hstart) nello stesso momento in cui il canale Dma Sprite le sta leggendo, è possibile (e succede sempre...) che lo Sprite si muova a scatti.

Si possono quindi cambiare le 2 words di controllo solo quando siamo sicuri che esse non siano lette dal canale Dma Sprite.

L'unico periodo in cui abbiamo la certezza che il canale Dma Sprite non le legga è durante il Vertical-Blank.

La tecnica utilizzata nel programmino di esempio riportata sull'*Hardware Reference Manual* (che assicura la modifica del contenuto delle 2 words durante il Vertical-Blank) è assurda, poichè fa sprecare troppo tempo al 68000; ne useremo un'altra.

La tecnica che useremo consiste nel far eseguire la routine di movimento (cioè quella che cambia le due words di controllo dello Sprite) ogni volta che ha inizio il Vertical-Blank.

Ciò è possibile in quanto (cfr. **"Scherzi di Colore", C.C.C. n. 69**) se è settato il bit 5 (Vertb) del registro Interno ($DFF09a) ogni volta che ha inizio il Vertical-Blank si avrà una richiesta d'Interrupt di livello 3.

Quindi basterà far puntare il vettore Interrupt di livello 3 ($6c) alla routine di movimento, ed il gioco è fatto, ricordandosi però che alla fine della routine di movimento il 68000 deve saltare all'indirizzo che era contenuto prima nel vettore d'Interrupt di livello 3.

Molti lettori avranno capito che per utilizzare facilmente gli Sprites serve una routine che permetta di posizionarli facilmente magari assegnando un solo dato.

La routine Pos

Questa routine è presente sia nel programma proposto che in quello pubblicato nello scorso numero.

Il compito è quello di ricalcolare, in base ai dati contenuti in un'apposita struttura (il cui indirizzo viene passato in A0), le nuove Words di controllo dello Sprite.

La struttura è la seguente:
X: 2 bytes
Y: 2 bytes
INDMEM: 4 bytes
MODELLO: 4 bytes
INDCOPPER: 4 bytes

Il primo campo dati contiene la X (senza dx), il secondo la Y (senza dy), il terzo l'indirizzo di memoria (in cui è stata allocata la struttura Dma Sprite), il quarto contiene l'indirizzo della sagoma dello sprite.

Il quinto campo dati indica dove si trovano le 2 istruzioni del Copper che ripristinano, ad ogni Vertical-Blank, il puntatore alla struttura Dma Sprite.

Come al solito, all'inizio della routine vengono salvati tutti i registri sullo stack.

E' da notare che salviamo **tutti** i registri, per questioni di "forma", benchè tale operazione sia sufficiente solo per 4 di essi (e cioè a1, d0, d1, d2).

Successivamente copiamo l'indirizzo della struttura Dma Sprite in A1, la X in d0 e d2 e la y in D1.

Dopo aver diviso per 2 la X, per ottenere i bits più significativi, sommiamo dx (= 64) a d0 e dy (= 44) a D1 e poniamo nella struttura Sprite Dma la nuova Vstart.

Poi ricaviamo Vstop tramite la relazione: Vstop = Vstart + N , dove N è l'altezza dello Sprite.

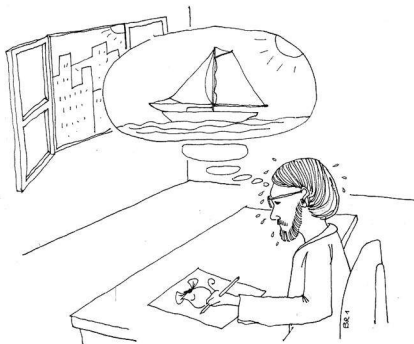
Volendo ottenere Sprite alti 5 pixels, N sarà 5. Ottenuta Vstop, la sistemiamo insieme a Hstart.

Successivamente ricaviamo lo stato che dovrà assumere il bit meno significativo di X e lo settiamo, o resettiamo, a seconda del caso. Infine togliamo tutto dallo stack.

L'Esempio Proposto

Il programma proposto, su cui si basava anche quello dell'articolo *Dorarge e la Cornacchia* (a cui rimandiamo per una descrizione più accurata) fa saltellare uno sprite lungo lo schermo.
Per far ciò sono state usate due tabelle, rispettivamente contenenti gli incrementi di X e di Y da aggiungere all'attuale posizione dello sprite. Cambiando i valori contenuti in tali tabelle potremo far eseguire vari percorsi.

```
; SPRITE BASE
; written
; by
; Donato De Luca

EXECBASE = $4
ALLOCMEM = -198
FREEMEM  = -210
OPENLIBRARY = -408
CLOSELIBRARY = -414
INTENA = $DFF09A
DMACON = $DFF096
CIAA  = $BFE001
VI3 = $6C
movem.l d0-d7/a0-a6,-(sp);Tutto Stack
move.l EXECBASE,a6 ;ExecBase->a6
lea GFXNAME,a1 ;Apro la
jsr OPENLIBRARY(a6)  ;Graphic
move.l d0,a5 ;In d0,non controllo
move.w #$0080,DMACON ;Copper bye bye.
move.l 50(a5),OLDCOPPER ;Salvo ind Sys CList
move.l #COPPERLIST,50(a5);Punto mia CList.
move.w #$8080,DMACON ;Resuscito Copper.

lea TABELLAMOSTRI,a2 ;Dove prendo SPRITE
move.w #0,d2 ;Quanti ( - 1 )
CHIEDIELEMOSINA: ;Alloco SPRITE in CHIP
move.l (a2)+,a3 ;Indirizzo SPRITE att.
move.l #$0002,d1 ;Voglio 35 bytes
move.l #7*5,d0  ;
jsr ALLOCMEM(a6) ;
move.l d0,MEMOGG ;
move.l 12(a3),a0 ;Copper seg sprite att
move.w d0,6(a0) ;LOW word
swap d0 ;
move.w d0,2(a0) ;HIGHT word
move.l MEMOGG,4(a3) ;Salvo ind reale SPR
move.l 8(a3),a0 ;Modello SPRITE
move.l MEMOGG,a1 ;Dove lo metto
move.l #35,d0 ;Quanti bytes
TRANLOOP:
move.b (a0)+,(a1)+ ;Copio Mod sprite mod
dbra d0,TRANLOOP ;
dbra d2,CHIEDIELEMOSINA ;Tutto per NSPRITE-1
move.l #320*256/8,d0 ;256 = PAL
move.l #$10002,d1 ;Chiedo la
jsr ALLOCMEM(A6) ;memoria pulita
move.l d0,PIANO1 ;
lea BITPLANE1,a0 ;Faccio puntare
move.w d0,6(a0)  ;il puntatore
swap d0 ;al piano
move.w d0,2(a0) ;appena preso
move.l #TABELLAX,TABXPOS ;Punto inizio tab X
move.l #TABELLAY,TABYPOS ;Punto inizio tab Y
move.w  #$8020,INTENA ;Attivo VERTB
move.l VI3,OLDIRQ ;Salvo vet IRQ3
move.l #NUCLEO,VI3 ;Nucleo-> vet IRQ3
MAIN: ;Aspetto finche` il
btst #6,CIAA ;Pulsante sin mouse
bne.s MAIN ;non viene premuto
EXIT:
move.w #0,d3 ;Quanti mostri?
lea TABELLAMOSTRI,a2 ;Dove sono le str?
BUCOLICO: ;Rido la memoria
move.l (a2)+,a3 ;a EXEC
move.l 4(a3),a1 ;d0=quantita`
move.l #35,d0 ;a1=indirizzo
jsr FREEMEM(a6) ;
dbra d3,BUCOLICO ;
move.l #320*256/8,d0 ;Rido il piano
move.l PIANO1,a1 ;
jsr FREEMEM(a6) ;
;rispristino vettore irq
move.l OLDIRQ,VI3 ;
move.w #$0080,DMACON ;Re:Bye Bye Copper
move.l OLDCOPPER,50(A5) ;Punto Old CList
move.w #$8380,DMACON ;Resuscito Copper.
move.l a5,a1 ;ind Graphic in A0
jsr CLOSELIBRARY(a6) ;chiudo la graphic.
movem.l (sp)+,d0-d7/a0-a6 ;Butto tutto fuori
rts ;Torna a casa,Pluto
NUCLEO:
movem.l D0-D7/A0-A6,-(SP);Stack in
lea MOSTRO1,A0 ;Struttura Sprite
move.l CONT,d0 ;Cont avanz tab dx e dy
cmp.l #20,d0 ;Sono a 20 incrementi?
bne NOFINETAB ;NO , no fine tab
FINETAB: ;Se sono arrivato alla
move.l #TABELLAX,TABXPOS ;fine tabella dei dx
move.l #TABELLAY,TABYPOS ;e dy rinizializzo
move.l #0,CONT ;puntatori alle tab
;e azzero cont.
NOFINETAB: ;
add.l #1,cont ;Inc 1 contatore dx,dy
move.l TABXPOS,a1 ;Ind tab dx in a1
move.w (a1)+,d0 ;Avanzo tab dx
add.w d0,(a0) ;Aggiorno X sprite
move.l a1,TABXPOS ;Salvo pos in tab dx
move.l TABYPOS,a1 ;Ind tab dy in a1
move.w (a1)+,d0 ;Avanzo tab dy
add.w d0,2(a0) ;Aggiorno Y sprite
move.l a1,TABYPOS ;Salvo pos in tab dy

jsr pos ;Ricalcolo Words di
;controllo della str.
;sprite dma
MOVEM.L (SP)+,d0-d7/a0-a6 ;Scarico
dc.w $4EF9 ;= JMP
OLDIRQ: ;Durante es prg
DC.L 0 ;metto ind a cui
 ;Jumpare
POS:
movem.l d0-d7/a0-a6,-(sp) ;Gnam Gnam
move.l 4(a0),a1 ;Ind sprite in mem
move.w (a0),d0 ;X in d0
move.w 2(a0),d1 ;Y in d0
move.w d0,d2 ;Copio d0
asR.w #1,d2 ;Div 2
add.b #64,d2 ;+ delta X
add.b #44,d1 ;+ delta y
move.b d1,(a1) ;VSTART
addq #5,d1 ;ricavo VEND
Move.b d1,2(a1) ;VEND
move.b d2,1(a1) ;Pari o dispari?
and.b #$01,d0 ;
or.w #0,d0 ;Lascio Word CTRL
move.b d0,3(a1) ;Salvo Word CTRL
movem.l (sp)+,d0-d7/a0-a6;
rts ;
OLDCOPPER: ;Puntatore alla
dc.l 0 ;vecchia copperlist

MEMOGG:
DC.L 0
TABELLAMOSTRI:
DC.L MOSTRO1
PIANO1:
DC.L 0
COPPERLIST:
dc.w $0180,$0000,$0182,$000f ;C0,C1 ( C=COLOR)
dc.w $01a2,$00f0,$01a4,$0fff
dc.w $01a6,$0f00 ;C 17,18,19
dc.w $008e,$2c81,$0090,$2cc1 ;DIWSTART,DIWSTOP
dc.w $0092,$0038,$0094,$00d0 ;DDFSTART,DDFSTOP
dc.w $0108,$0000,$0102,$0000 ;MODULO,SCROLL
```

```
BITPLANE1:
dc.w $00e0,$0005,$00e2,$0000 ;BPL0POINTERS
dc.w $0100,$1200  ;un bitplane (320*255)
SPRITEPTR0:
dc.w $0120,$0001,$0122,$8000 ;1(0) DMA
SPRITEPTR1:
dc.w $0124,$0001,$0126,$9000 ;2(1) DMA
SPRITEPTR2:
dc.w $0128,$0001,$012a,$9000 ;3(2) DMA
SPRITEPTR3:
dc.w $012c,$0001,$012e,$9000 ;4(3) DMA
SPRITEPTR4:
dc.w $0130,$0001,$0132,$9000 ;5(4) DMA
SPRITEPTR5:
dc.w $0134,$0001,$0136,$9000 ;6(5) DMA
SPRITEPTR6:
dc.w $0138,$0001,$013a,$9000 ;7(6) DMA
SPRITEPTR7:
dc.w $013c,$0001,$013e,$9000 ;8(7) DMA
dc.w $ffff,$fffe      ; End Copper List

SPRITE:
dc.w $9040,$9500
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0000,$0000
MOSTRO1:
DC.W 0 ;X
DC.W 56 ;Y
DC.L 0 ;MEM RELAE
DC.L SPRITE ;MODELLA
DC.L SPRITEPTR0 ;SEG COPPER PRG.
CONT:
DC.L 0
TABELLAX:
DC.W 1,1,1,1,2,2,2,3,3,4
DC.W 4,3,3,2,2,2,1,1,1,1
TABELLAY:
DC.W 1,1,1,1,2,2,2,3,3,4
DC.W -4,-3,-3,-2,-2,-2,-1,-1,-1,-1
TABXPOS:
DC.L 0
TABYPOS:
DC.L 0
even
GFXNAME:
dc.b 'graphics.library',0
```

UN VELOCISSIMO SPARA-IMMAGINI

Sono disponibili numerose routine per far apparire, i n propri programmi, schermate di presentazione in formato IFF;quella che presentiamo, però, ha il dono della super-velocità

di Stefano Silvestri

Come è ormai universalmente risaputo, nel microcosmo informatico di Amiga, quando si parla di grafica si parla invariabilmente di **standard IFF**.

Ormai tutti, o quasi, sanno che cosa significhi l'acronimo (Interchange Format Files), ma pochi (o per mancanza di tempo, o per mancanza di costanza o per problemi di lingua), approfondiscono l'argomento.

I primi tempi in cui si ha a che fare con la tastiera (addio Joy, compagno di tante partitone a Kick Off), e con il set di istruzioni del 68000, sembra di essere dei perfetti imbecilli, incapaci di esprimere il più semplice concetto logico.

Inizia a bloccarsi la salivazione, improvvisamente nella nostra mente si insinua un costante rumore di fondo, che copre le eventuali già scarse idee e gli addirittura rarissimi concetti.

A questo punto il 90% dei neoprogrammatori vanno in pensione: riprendono a giocare a Kick Off (grazie Anco...), e se vogliono programmare qualcosa, si lasciano ammaliare dalla rassicurante sintassi del Basic.

Che per altro spesso, soprattutto oggi, si rivela perfettamente all'altezza della situazione, a patto di non chiedergli cose turche (avete dato un'occhiata al GFA BASIC?).

Ciò non significa che, per dilettarsi a programmare in Assembler, si debba avere un QI elevatissimo.

Semplicemente basterebbe, per esempio, che i *Testi Sacri* venissero tradotti nella nostra lingua, che la maturità informatica del nostro Paese fosse un fatto acquisito... e via di questo passo utopisticamente.

Tutti, o quasi, potrebbero dilettarsi con l' Assembler. Invece siamo ancora qui a spiegare cosa sia la gameport-1.

Ma passiamo al sodo e parliamo dello standard IFF.

Lo standard IFF

L'IFF è uno standard che permette di utilizzare un formato unico per la gestione dei dati. In questo modo programmi diversi possano scambiarsi dati senza ricorrere a particolari soluzioni hardware o software, evitando la necessità di una miriade di formati incompatibili. Parliamo di dati (in generale) e non solo di dati grafici perchè i dati possono essere di natura diversa: **Grafici, Sonori, Testo,** ecc.

In queste pagine si tratterà, nella fattispecie, di file Grafici, la cui struttura (come i files prodotti da **Dpaint III**) è molto semplice.

Essa è realizzata a blocchi (**chunks**), ed ogni blocco assolve ad una funzione particolare. La struttura dei blocchi è la seguente:

4 bytes per il tipo di blocco (es: CMAP BODY), 4 bytes che ne definiscno la lunghezza (LEN) LEN bytes... contenenti informazioni diverse a seconda del tipo di blocco.

I blocchi più importanti sono:

ILBM: informa che il file è di tipo **I**nter**L**eaved-**B**it**M**ap.

BMHD: il **B**it**M**ap**H**ea**D**er fornisce informazioni riguardanti Altezza, Larghezza, Profondità, tipo di compressione, maschere eventualmente usate ecc. .

CMAP: la **C**olor**MAP**, invece, fornisce informazioni per quel che riguarda la tavolozza dei colori da usare per rappresentare l'immagine.

CAMG: assolve ad una sola funzione, per la verità fondamentale, definendo il *modo* di visualizzazione da usare.

BODY: in inglese significa *corpo*, ed infatti sta ad indicare l'inizio dei dati grafici veri e propri, quelli che andranno scritti sullo schermo, o me-

glio, nei vari BitPlanes. Il listato presentato in queste pagine è un lettore di files IFF grafici, che non richiede, per il suo corretto funzionamento, la presenza di alcun file o directory particolare nel disco dove si trova. La sintassi di chiamata, se il programma lo registrate con il nome **Fiff**, è:

[dfn]. [dir]. fiff [dfn]. [dir]. nomefile

Ovviamente **nomefile** dovrà essere un file in formato IFF (ILBM), e non dovrà superare i 20 caratteri. (Questo limite è però facilmente modificabile). Omettendo, per dimenticanza, il nome del file, può verificarsi un blocco del sistema.

In caso di errore "generico" il programma non viene eseguito e viene ridato il controllo all'utente del CLI. Il listato possiede le seguenti caratteristiche:

1) E' in grado di leggere tutti i modi di risoluzione standard di Amiga: **Ham, HalfBrite, Lo-Res Hi-Res, Interlace**.

2) Occupa poca memoria.

3) E' decisamente veloce.

Per quanto riguarda il primo punto, il blocco CAMG è così costituito:

CAMG... nome del blocco
0004... lunghezza dei dati in bytes
0000... dati

Vi sono 4 bytes dati, cioè una longword della quale bisogna ignorare la word più significativa e scrivere solo quella meno significativa nel campo **viewmodes** della struttura screen.

I punti 2 e 3, invece, devono essere trattati insieme per realizzare un lettore veloce. Basta leggere l'intero file in una volta sola, decomprimerlo in memoria e visualizzarlo. Il metodo descritto, però, per un file di 30k a 32 colori in LO-Res richiede circa 50k di memoria video e 30k di memoria come buffer temporaneo per la lettura del file.

Dal momento che Amiga dispone di 512k, ma li succhia come un'aranciata, appena vorrete realizzare qualcosa di dimensioni "visibili", vi renderete conto che 30k sono preziosi e capirete che cosa significano tanti colori su schermo, (colori fa rima con dolori).

D'altra parte è facile anche realizzare un lettore che non mangia kappa su kappa come noccioline.

Sarà sufficiente leggere, dal file su disco, una longword alla volta, tradurla, agire di conseguenza e passare alla successiva longword.

Il consumo di memoria, in questo caso, risulta il minimo indispensabile. Se lanciamo il programma ci accorgiamo che occorrono circa 25 secondi per caricare una semplice schermata... roba da registratore a cassette! E allora?

Poniamoci una domanda (retorica naturalmente): che cosa rende il nostro Amiga tanto lento?: l'accesso alle periferiche, e per essere più precisi, il movimento meccanico della testina. Il nocciolo della questione sta nel ridurre il numero di accessi al disco senza allocare troppa memoria.

Chi non sa che cosa sia un **buffer** o lo impara ora o mai più. Il buffer è nato ancor prima del computer; quotidianamente ne facciamo uso. Il postino, ad esempio, dispone di un sacco (buffer) pieno di lettere da consegnare.

Sarebbe pazzo se uscisse dall'ufficio postale con una lettera da consegnare alla volta, perchè farebbe più fatica per ottenere un risultato notevolmente inferiore a quello ottenibile con il sacco-buffer.

Un buffer è quindi una zona di memoria *protetta*, dove un programma qualunque, che abbia la necessità, può parcheggiare temporaneamente dei dati, per poi trattarli a blocchi. Migliorano così le prestazioni velocistiche di periferiche e subroutine in generale.

Il listato

Il listato usa un semplice, vecchio, banalissimo buffer da 128 bytes, dove vengono memorizzati temporaneamente i tronconi del file da leggere per essere decompressi.

Una volta giunti alla fine del buffer si accede al disco nuovamente per leggere altri 128 bytes e così via, fino alla fine del file. In questo modo se, prima, un file da 30k richiedeva circa 30000 accessi al disco, ora il numero di accessi è ridotto a circa 230.

Il risultato è un tempo di caricamento, per lo stesso file di prima, di appena 3.5 secondi, con un consumo extra di memoria di appena 128 bytes. Ovviamente il buffer può essere facilmente aumentato o diminuito a proprio piacimento con minime modifiche nel listato. E' da notare che già con un buffer di soli 32 bytes la velocità aumenta, mentre usare buffer di oltre 1024 bytes non comporta ulteriori miglioramenti. Il listato può essere usato in un semplice batch-file come presentazione, oppure all'interno di un programma, più complesso, come subroutine di caricamento di schermate.

Inoltre, dal momento che richiede l'apertura di una finestra, sarà possibile utilizzarne i gadgets. Ecco uno schema dei blocchi e delle loro funzioni generiche:

1) Opzioni di compilazione.
Inclusioni varie.

2) Apertura delle librerie necessarie e controllo loro validità.

Apertura del file e allocazione della memoria di buffer.

3) Lettura file e relativi blocchi ILBM BMHD CMAP CAMG, settaggio delle variabili nelle strutture di schermo, finestra, colormap ecc.
4) Apertura schermo, finestra, settaggio colori.
5) Decodifica dei dati grafici veri e propri e loro scrittura.
6) Ciclo di ritardo di esempio (qui potete continuare voi).
7) Chiusura librerie, schermi, finestre e deallocazione di memoria.
8) Subroutines di lettura dati da disco.
9) Sezione dati. Definizioni di costanti e strutture.

```
        opt l-,d+
* Nella prossima riga usare ram: df1: o df0: secondo delle proprie esigenze
        incdir "ram:include/"
        include "libraries/dos_lib.i"
        include "libraries/dos.i"
        include "exec/exec_lib.i"
        include "exec/memory.i"
        include "graphics/text.i"
        include "intuition/intuition.i"
        include "intuition/intuition_lib.i"
        include "graphics/graphics_lib.i"
        subq #1,d0                          D0-1 =lungh effettiva
        move.l #nomefil,a1                  adress nomefil in a1
cont    move.b (a0)+,(a1)+                  copio da (a0) ad (a1)
        subq #1,d0                          contatore-1..
        bne cont                            fine riga comando?
        move.l _SysBase,a6                  SysBase in a6
        movea.l #dosname,a1                 nome.library
        moveq #0,d0                         versione indiff
        jsr _LVOOpenLibrary(a6)             chiamata
        beq fineprg                         controllo risultato
        move.l d0,dosbase                   salvo risultato
        movea.l #intname,a1                 apro intuition..
        moveq #0,d0
        jsr _LVOOpenLibrary(a6)
        beq finedos
        move.l d0,intbase
        movea.l #gfxname,a1                 apro graphics..
        moveq #0,d0
        jsr _LVOOpenLibrary(a6)
        beq fineint
        move.l d0,gfxbase
        move.l #nomefil,d1                  metto nomefile in d1
        move.l #MODE_OLDFILE,d2             modo di accesso
        movea.l dosbase,a6                  base del dos in A6..
        jsr _LVOOpen(a6)                    chiamata
        beq finegfx                         se errore finisco
        move.l d0,fhin                      filehandle(d0) in FHIN


        move.l #128,d0                      voglio 128 bytes
        move.l MEMF_CHIP|MEMF_CLEAR,d1      tipo CHIP e CLR
        move.l _SysBase,a6                  SysBase in a6
        jsr _LVOAllocMem(a6)                chiamta
        beq clo.fil                         se = 0 salto
        move.l d0,fastload                  salvo ptr.
        moveq #8,d3                         leggo 8 bytes..
        bsr leggo                           chiamata
cilbm   moveq #4,d3                         leggo 4 bytes..
        bsr leggo                           chiamata
        cmpi.l #'ILBM',basemem              comparo con ILBM
        bne cbmhd                           se <> salto
        bsr leggo                           chiamata
cbmhd   cmpi.l #'BMHD',basemem              comparo con BMHD
        bne ccmap                           se <> salto
        bsr leggo                           leggo len di BMHD
        move.l basemem,d3                   metto len in d3
        bsr leggo                           leggo d3 bytes
        move.l #basemem,a3                  metto adress di
```

```
        move.w (a3),largw           tavola di BMHD
        move.w (a3)+,largs          in a3 poi vi
        move.w (a3),altw            prelevo i dati
        move.w (a3)+,alts           con il metodo a
        move.w (a3)+,left           postincremento
        move.w (a3)+,top            e li metto nelle
        move.b (a3)+,dept           apposite strutture
        moveq #0,d4
        moveq #4,d3                 leggo 4 bytes..
        bsr leggo                   chiamata
ccmap   cmpi.l #'CMAP',basemem      comparo con CMAP
        bne ccamg                   se <> salto
        bsr leggo                   leggo 4 bytes(len)
        cmpi.l #96,basemem          piu di 32 colori?
        ble nohb                    se <=32 salto
        sub.l #96,basemem           tolgo colori in piu'
        move.l #$f,d4               d4 usato come flag
nohb    move.l fhin,d1              FHIN in d1
        move.l #coltab,d2           in coltab metto i dati
        move.l basemem,d3           metto len in D3
        jsr _LVORead(a6)            chiamata..
        move.l #coltab,a2           registro per output dati
        movea.l #coltab,a5          registro per input dati
cicla   moveq #0,d5                 pulisco d5
        move.b (a5)+,d5             in d5 byte red e
        lsl.w #4,d5                 un nibble a sin..
        move.b (a5)+,d5             in d5 byte green e
        lsl.w #4,d5                 un nibble a sin..
        move.b (a5)+,d5             in d5 byte green e
        lsr.w #4,d5                 un nibble a dex..
        move.w d5,(a2)+             metto dati colore in a2+
        sub.l #3,d3                 decr d3 di 3
        bne cicla                   ciclo se <>0

        tst.l d4                    se d4 <> 0
        bne seek                    salto a seek

        moveq #4,d3                 leggo 4 bytes..
        bsr leggo                   chiamata
ccamg   cmpi.l #'CAMG',basemem      comparo .'CAMG'
        bne cbody                   se <> 0 salto
        bsr leggo
        bsr leggo                   chiamata
        move.w basemem+2,mode       mode 0 = tipo screen

        bsr leggo                   leggo 4 bytes
cbody   cmpi.l #'BODY',basemem      comparo.'BODY'
        beq aproscr                 se = 0 salto

        bsr leggo                   leggo 4 bytes
seek    move.l fhin,d1              fhin in d1
        move.l basemem,d2           offset di spostam.
        move.l #OFFSET_CURRENT,d3   modo di ricerca
        jsr _LVOSeek(a6)            chiamata
        bra cilbm                   salta sempre
aproscr moveq #4,d3                 leggi i 4 bytes
        bsr leggo                   di BODY SIZE
        lea  screen,a0              adress screen in a0
        move.l intbase,a6           intuibase in a6
        jsr _LVOOpenScreen(a6)      chiamata
        beq fremem                  se = 0 salto
        move.l d0,p.screen1         risult. in p.screen1
        move.l p.screen1,p.screenf1 ptr.screen->p.screenf1
        move.l p.screen1,a0         ptr.screen in a0
        add.l #44,a0                a0 + offset = viewport
        moveq #32,d0                numero colori
        move.l #coltab,a1           adress  color map
        move.l gfxbase,a6           graficlib in a6
        jsr _LVOLoadRGB4(a6)        chiamata
aprowin lea window,a0               ptr.window->a0
        move.l intbase,a6           baseintu in a6
        jsr _LVOOpenWindow(a6)      chiamata
        beq clo.scr                 se = 0 salto
        move.l d0,p.window1         risult. in p.window1
```

```
        move.l p.screen1,a5          punt.screen1 in a5
        add.l #192,a5                aggiungo offset
        move.l a5,vettorebp          salvo adress base bitmap
        move.l (a5),a5               metto a5-> in a5

        move.w largs,d7              larg in d7 in pixel
        lsr.l #3,d7                  trasformata in bytes
        move.w d7,bpl                e salvata in bpl
        bsr read128                  leggo primi 128 bytes
        moveq #0,d5                  d5 = y
        moveq #0,d6                  d6 = bitplane
        moveq #0,d7                  d7 = bytes
loopbp  move.w bpl,d7                metto larg in d7
        mulu.w d5,d7                 y attuale * bpl
        lsl.w #2,d6                  ricavo offset
        add.l vettorebp,d6           vettorebp + offset
        move.l d6,a5                 d6 in a5
        move.l (a5),a5               a5 = valore punt. da a5
        add.l d7,a5                  aggiungo offset
        moveq #0,d7                  clear d7
        sub.l vettorebp,d6           rimetto d6 nella
        lsr.w #2,d6                  condizione iniziale
loopx   moveq #0,d4                  clear d4
        move.b (a4)+,d4              metto valore letto in d4
        sub.l #1,contab              contba -1
        bne step                     se <> 0 salto
        bsr read128                  ciamata
step    cmp.b #128,d4                lo comparo con 128..
        bhi max                      e salto a maggiore..
        bmi min                      a.. minore....
        bra loopx                    o lo ignoro (codice 128)
min     add.b d4,d7                  ..d7 conta bytes
        addq #1,d7                   bytes da ricopiare+1
rep     move.b (a4)+,(a5)+           da buffer a mem video
        sub.l #1,contab              cont di buffer -1
        bne step1                    se <> 0 salto
        bsr read128                  leggo 128 bytes nuovi
step1   dbf d4,rep                   ripeto finche' d4 =-1
        bra ctr.x                    salto a ctr.x
max     neg.b d4                     d4 negato=num ripet.
        add.w  d4,d7                 aggiorno d7 cont.byte
        addq #1,d7                   correggo d7
ciclo   move.b (a4),(a5)+            dato in a5->
        dbf d4,ciclo                 ripeto finche' d4 =-1
        addq #1,a4                   incr. ptr. a buffer
        sub.l #1,contab              cont di buffer -1
        bne ctr.x                    se <> 0 salto
        bsr read128                  leggo 128 bytes nuovi
ctr.x   cmp.w bpl,d7                 letta un riga?
        bmi loopx                    se no salto
        moveq #0,d7                  clr contabytes
ctr.bp  addq #1,d6                   +1 bitplanes..
        cmp.b dept,d6                letti tutti bitplanes?
        bmi loopbp                   se no salto
        moveq #0,d6                  clr bitplanes
ctr.y   addq #1,d5                   +1 riga(y)
        cmp.w alts,d5                lette tutte righe?
        bmi loopbp                   se no salto
*********************************************************************
* Da qui potete iniziare a fare le vostre aggiunte..come l'esempio
* orrido sotto riportato...
        move.l #1000000,d5           loop di ritardo
ritardo sub.l #1,d5                  di esempio
        bne ritardo                  molto rozzo
*********************** ROUTINES DI CHIUSURA *********************

clo.win move.l p.window1,a0          in a0 punt. p.window1
        move.l intbase,a6            intuibase in a6
        jsr _LVOCloseWindow(a6)      chiamo la funz.
clo.scr move.l p.screen1,a0          in a0 il punt. p.screen1
        move.l intbase,a6            intuibase in a6
        jsr _LVOCloseScreen(a6)      chiamo la funz.
fremem  move.l #fastload,a1
        move.l #128,d0
```

```
        jsr _LVOFreeMem(a6)
clo.fil move.l  fhin,d1                     fhin in D1
        move.l  dosbase,a6                  dosbase in A6
        jsr _LVOClose(a6)                   chiamo la funz.
finegfx move.l gfxbase,a1                   in a1 graficbase
        move.l  _SysBase,a6                 in a6 SysBase
        jsr _LVOCloseLibrary(a6)            chiamo la funz.
fineint move.l intbase,a1                   in a1 intuibase
        move.l  _SysBase,a6                 in a6 SysBase
        jsr _LVOCloseLibrary(a6)            chiamo la funz.
finedos move.l dosbase,a1                   in a1 dosbase
        move.l  _SysBase,a6                 in a6 SysBase
        jsr _LVOCloseLibrary(a6)            chiamo la funz.
fineprg rts
************************** SUBROUTINES *************************
leggo   move.l dosbase,a6           dosbase in a6
        move.l fhin,d1              filehandle in d1
        move.l #basemem,d2          basemem in d2(dato sara' in d2)
        jsr _LVORead(a6)            chiamata
        rts
read128 move.l dosbase,a6           dosbase in a6
        move.l fhin,d1              filehandle in d1
        move.l fastload,d2          fastload in d2(dato sara' in d2)
        move.l #128,d3
        jsr _LVORead(a6)            chiamata
        move.l #128,contab
        move.l fastload,a4
        rts
************************** SEGMENTO DATI ************************
        cnop 0,2
nomefil         ds.b    20
dosname         dc.b    'dos.library',0
intname         dc.b    'intuition.library',0
gfxname         dc.b    'graphics.library',0
dosbase         dc.l    0
intbase         dc.l    0
gfxbase         dc.l    0
fhin            dc.l    0
p.screen1       dc.l    0
p.window1       dc.l    0
vettorebp       dc.l    0
fastload        dc.l    0
contab          dc.l    0
basemem         ds.b    104
coltab          equ     basemem+8
bpl             dc.w    0
        cnop 0,2
screen
left            dc.w    0
top             dc.w    0
largs           dc.w    320
alts            dc.w    256
pad             dc.b    0
dept            dc.b    5
                dc.b    2,4
mode            dc.w    0
                dc.w    CUSTOMSCREEN
                dc.l    0
                dc.l    0
                dc.l    0
                dc.l    0
window
                dc.w    0,0
largw           dc.w    0
altw            dc.w    0
                dc.b    1,2
idcmp           dc.l    0
flag            dc.l    BORDERLESS
                dc.l    0,0
                dc.l    0
p.screenf1      dc.l    1
                dc.l    0
                dc.w    0,0,0,0
                dc.w    CUSTOMSCREEN
```

AMIGABASIC CHIAMA SEKA

I primi passi in Assembly è bene farli.. in Basic!

di Gabriele Bellussi

Il programma proposto questo mese rappresenta un valido aiuto per chi scrive brevi routine in Assembler e vorrebbe vederle girare anche in Basic, con poca fatica. Il programma è scritto interamente in Basic e non presenta nessuna routine in Linguggio Macchina al suo interno.

Per questo motivo si consiglia (soprattutto ai principianti) di studiarlo attentamente. Lo scopo è quello di rendere eseguibili da basic le routine che normalmente possono girare o su assemblatori (in questo caso **K-Seka**) o in **ambiente CLI**.

Il programma funziona, però, solo con routine in **formato oggetto**. Assieme al programma basic è presente una routine Assembler con l'unico scopo di collaudare il programma una volta digitato e lanciato, perciò è consigliabile digitare prima la routine, salvarla con il comando **WO** chiamandola ad esempio...

:routine.prova

Per chi ancora non lo sapesse, il comando **WO** serve per salvare la routine Assembler in formato **oggetto**, mentre il comando **W** salva la routine in formato **sorgente** sotto forma di file Ascii. Entrate quindi in ambiente basic, digitate il programma proposto e salvatelo prima di mandarlo in esecuzione con il Run.

Appare il *primo* messaggio che chiede l'inserimento del nome della routine Assembler (in questo caso si dovrà digitare **:routine.prova**), il *secondo* messaggio chiede l'inserimento del nome della routine con cui vorremo lanciarla da basic (ad esempio **:routine.basic**). Una volta inserito il nome si sentirà il drive (o i drives) rosicchiare il disco e alla fine apparirà l'**ok** del basic.

A questo punto il grosso delle operazioni è finito, non resta che cancellare con un New il programma e caricare **:routine.basic** con il comando **Load** per poi mandarlo in esecuzione con **Run**.

Sorpresa! lo schermo si è riempito di righe colorate che si succedono alternativamente per creare sfumature di colore che vanno dal giallo al rosa fino al viola intenso (e poi dicono che i computer rovinano la fantasia dei ragazzi).

Per bloccare la routine basta la pressione del pulsante sinistro del mouse.

Provando a listare il programma che genera l'effetto cromatico si noterà una serie di linee **Data** contenenti valori esadecimali i quali non sono altro che i codici macchina che compongono la routine L.M.

In fondo alla routine è presente il **caricatore** che la legge, la memorizza nel vettore **routine%** (dopo aver convertito i codici in base decimale) e la manda in esecuzione con il comando **CALL** (vedi C.C.C. n. 70). Ma come è possibile tutto ciò?

❑❒

La tecnica

L'Amiga è un computer in grado di lavorare a 16 (word) e a 32 (long word) bits, ma per ottenerli il computer deve affiancare più bytes per volta, quindi per ottenere una word verranno affiancati 2 bytes mentre per la long word saranno necessari 4 bytes.

Le istruzioni dell'Assembler (**move, jmp, rts** eccetera) sono tutte formate da un codice che occupa un'intera word, ma se si provasse ad entrare nel monitor del linguggio macchina (per i possessori del **Seka** assembler basterà digitare **M [indirizzo]**) si noterà che i codici che compaiono sullo schermo sono tutti ad otto bits. Ad esempio, si provi a digitare, in Seka, il seguente, breve programmino:

esempio: rts

Una volta assemblato, si provi a digitare...

M esempio

In questo modo siamo *entrati* nel monitor e ciò che appare sono i codici contenuti nelle memorie del computer all'interno delle quali c'è anche il nostro esempio. Il codice macchina dell'istruzione **Rts** è **$4e75** ed è un codice a 16 bits, ma all'interno della lista apparsa dopo il comando *m esempio* esso appare come **$4e** e **$75**, cioè è stato diviso in **due** codici a 8 bits che, affiancati, danno come risultato una word.

Un altro esempio. L'istruzione...

move.w indirizzo,indirizzo

...ha come codice macchina il valore $33f9; quindi l'istruzione:

move.w $dff182,$dff180

...verrà memorizzata nel seguente modo:

33, f9, 00, df, f1, 82, 00, df, f1, 80

Quando si vuole salvare su disco una routine Assembler in formato oggetto, il computer salverà i singoli codici ad 8 bit che la compongono trasformandoli in caratteri secondo il codice Ascii.

Riferendoci all'esempio precedente, il primo codice macchina (**$33**) verrà salvato su disco come carattere "**3**" (il codice Ascii del carattere "3" è $33) e così via per tutti gli altri.

Precedentemente era stato consigliato di digitare la routine Assembler proposta chiamandola *routine.prova* e di salvarla in formato oggetto; adesso provate a digitare...

type:routine.prova

Una volta impartito il comando, sullo schermo apparirà una serie di caratteri particolari accompagnati da vari *beep* e questa è la dimostrazione pratica di quanto detto fino ad ora (anche i beep sono caratteri speciali).

E' molto importante sapere come il computer salva le routine su disco perchè aiuterà a comprendere il programma proposto.

Questo non fa altro che caricare la routine in memoria, prelevare i singoli caratteri che la compongono e ricavare, da questi, i codici macchina della routine.

Il programma apre due files, uno in input e l'altro in output; nel primo viene caricata all'interno della variabile A$ la routine e nel secondo viene scritta su disco la routine eseguibile da basic (con relativo caricatore).

Una volta caricata la routine, viene aperto un ciclo For Next nel quale si possono distinguere due sezioni (contraddistinte dalle *REM sezione 1* e *sezione 2*) le quali hanno la stessa funzione: la prima preleva un carattere dalla variabile A$, ne estrae il codice Ascii e lo trasforma in un codice esadecimale memorizzandolo nella variabile B$; in seguito si salta alla subroutine *completa* (della quale parleremo più avanti) per passare poi alla seconda sezione, che estrae dalla variabile A$ il carattere successivo, ne determina il codice Ascii e lo trasforma in base esadecimale memorizzandolo nella variabile **Routine$**, infine salta alla subroutine *completa*. Terminate queste due operazioni le due variabili vengono affiancate all'interno della variabile **Routine1$** per passare alla fase successiva nella quale vengono create le **linee Data** che conterranno i codici esadecimali della routine Assembler e salvate su disco in forma Ascii.

Ciò che scaturirà dal programma proposto sarà quindi un altro programma, sempre in basic, contenente la routine (in questo caso ":routine.prova") richiamabile da basic (con il comando **Load**) oppure inseribile in altro programma (con la funzione **Merge**).

Per spiegare la funzione di vitale importanza svolta dalla subroutine *completa*, sarà necessario ricorrere ad un esempio.

Immaginiamo che nel programma proposto sia assente la subroutine *completa* e che si debbano caricare da disco due caratteri che, affiancati tra loro, diano origine al valore **$D00F**.

Per fare ciò il programma trasformerà i rispettivi codici Ascii da decimale ad esadecimale e li memorizzerà in due variabili stringa, successivamente affiancate per dare origine al codice $d00f.

In realtà non è così perchè l'istruzione Basic **Hex$** non considera, ovviamente, gli zeri *di troppo*, perciò il programma non darà come risultato **$d00f** ma **$d0f** perchè il primo valore è formato dai codici $d0 e $0f, il secondo è formato dai codici $d0 e $f che sono uguali se vengono considerati come due codici separati ma se vengono affiancati danno origine a due valori differenti ($d00f <> $d0f).

Stessa sorte subiranno, ad esempio, i valori $0ffg, $000f, $df00 e $0f00 che verranno trasformati rispettivamente in: $ffg, $0f, $df0 e $f0. Per evitare che ciò accada la subroutine verifica se la variabile B$ è lunga due caratteri. Se la verifica è positiva si ritorna al programma principale, altrimenti aggiunge **$0** all'inizio della variabile **B$**.

La routine

Adesso è d'obbligo parlare della routine Assembler che tanto gentilmente si è offerta(!) come cavia per il nostro piccolo esperimento. Dal punto di vista puramente tecnico non presenta sostanzialmente nulla di eccezionale. Le istruzioni che la compongono (move, cmp, bhi, add) sono già state presentate nei numeri scorsi di C.C.C. tranne una.

Nel listato è infatti presente l'istruzione **Clr** che ha come funzione quella di azzerare una loca-

La tecnica descritta assomiglia molto a quella, nota ai 64-isti, che allocava routine l.m. grazie ai comandi Read e Data

zione o un registro (Clr=CLeaR). Tale istruzione ha tre sintassi fondamentali: la prima è **Clr.B** (o semplicemente Clr) che azzera solo il primo byte, la seconda è **Clr.W** e azzera i primi due byte (word) e l'ultima è **Clr.L** che azzera i primi quattro byte (long word). Al suo posto si potrebbe benissimo utilizzare l'istruzione Move digitando...

Move.W $0000,indirizzo

...ma l'uso dell'istruzione Clr è più conveniente e più semplice. Un'altra sintassi del comando Clr è la seguente:

Clr.B (A0)
Clr.W (A0)
Clr.L (A0)

...tramite la quale è possibile puntare alla locazione contenuta in A0, e quindi azzerarla. La routine proposta, con la sua prima riga, disabilita gli interrupts, mentre la seconda disabilita i canali DMA (vedi C.C.C. #69); la riga successiva azzera il registro D0 per passare poi al ciclo vero e proprio nel quale D0 viene incrementato di $1, viene confrontato il valore contenuto in D0 con $ffe e se il primo è minore del secondo allora il contenuto di D0 viene trasferito in $dff180 altrimenti si salta all'inizio della routine. Se si avvera la prima condizione, oltre a trasferire il valore di D0 in $dff180, il computer controlla se è stato premuto il tasto sinistro del mouse.

```
' ConvertSEKA
' rende eseguibili da Basic
' routine in Assembler
' by Gabriele Bellussi
CLS
zv = 0
LOCATE 12, 10: INPUT "nome routine:";file$
LOCATE 14, 10: INPUT "nome conversione:";file1$
OPEN "o", #2, file1$: 'Apre file nel quale salvare la routine elaborata
OPEN file$ FOR INPUT AS 1: 'Legge la
a$ = INPUT$ (LOF (1), 1): 'routine in
CLOSE 1: 'Assembler
' la routine Assembler viene elaborata
' sezione 1
FOR i = 1 TO LEN (a$)
s$ = MID$ (a$, i, 1)
a = ASC (s$)
b$ = HEX$ (a)
GOSUB completa
routine$ = b$
i = i + 1
' sezione 2
s$ = MID$ (a$, i, 1)
a = ASC (s$)
b$ = HEX$ (a)
GOSUB completa
routine1$ = routine$ + b$
' fine sezione 2
' Salva codici in linee DATA
z = zv/4
IF z = INT (z) THEN PRINT#2, linea$: linea$ = "data " + routine1$: zv = zv + 1: GOTO loop
linea$ = linea$ + ", " + routine1$: zv = zv + 1
loop:
NEXT i
' Salva il caricatore in coda alle linee DATA
zv$ = STR$ (INT (i/2)-4)
PRINT#2, "dim routine% (" + zv$ + ")"
PRINT#2, "for i = 1 to" + zv$
PRINT#2, "read routine$"
PRINT#2, "routine% (i) = val (" + CHR$ (34) + "&h" + CHR$ (34) + " + routine$)"
PRINT#2, "next i"
PRINT#2, "routine& = varptr (routine% (1))
PRINT#2, "call routine&"
CLOSE 2
END
'subroutine "completa"
completa:
IF LEN (b$) = 2 THEN torna
b$ = "0" + b$
torna:
RETURN
```

```
inizio:
        move.w #$4000,$dff09a    ; disattiva interrupt
        move.w #$0200,$dff096    ; disattiva canali DMA
        clr.w d0                 ; azzera il registro D0
ciclo:
        add.w #$1,d0             ; incrementa D0 di $1
        cmp #$ffe,d0             ; e confronta D0 con $ffe
        bhi inizio               ; se D0 maggiore di $ffe vai a inizio
        move.w d0,$dff180        ; altrimenti trasferisci D0 in $dff180
        btst #6,$bfe001          ; tasto sinistro mouse premuto ?
        bne ciclo                ; se no vai a ciclo
        move.w #$c000,$dff09a    ; altrimenti riattiva interrupt
        move.w #$8200,$dff096    ; e canali DMA , infine esci
rts
```

Amiga Action Replay

Finalmente! Una potentissima cartuccia utility+freezer+trainer!

Inserita nella porta di espansione del vostro Amiga 500, permette di:

- congelare e salvare su disco un programma caricato in memoria, per poterlo ricaricare quando volete fino a 4 volte più velocemente
- trovare le "poke" necessarie per ottenere vite infinite nei vostri giochi preferiti
- modificare e cambiare gli sprites di un gioco, per creare simpatiche versioni personalizzate o usare gli sprites nei vostri programmi
- avvertire della presenza di qualsiasi virus in memoria o sui vostri dischetti, distruggendo tutti i virus conosciuti
- salvare schermate e musiche su disco come files IFF, per poterle elaborare dai vostri programmi preferiti
- rallentare lo svolgimento dei giochi fino al 20% della velocità originale, per aiutarvi negli schermi più complicati
- usare il più potente monitor-disassembler per Amiga, con completo controllo dell'hardware e dei suoi registri (anche quelli "write-only"), uno strumento preziosissimo per il debugging dei vostri programmi: screen editor, breakpoint dinamici, assembler/disassembler delle istruzioni Copper, disk I/O con possibilità di alterare parametri quali sync o lunghezza della traccia, calcolatrice, notepad, ricerca di immagini o suoni in tutta la memoria, modifica caratteri in memoria, altera i registri della CPU, ed altro ancora.

Amiga Action Replay originale con manuale *in italiano* a sole *179.000*

ACCESSORI

AMAS Sound Digitizer 299.000
Hard disk A-590 899.000
Espansione 2 MB per A-590 399.000
Mac-2-DOS con drive 950.000
Espansione 2 MB A-2000 799.000
DigiDroid 175.000
DigiView 4.0 450.000
Drive esterno con switch 179.000
Drive esterno TrackDisplay 259.000
Drive esterno 5"1/4 275.000
Flicker Fixer 950.000
Scanner A4 1.495.000

HARDWARE

Espansione da 2 MB per A-500, si inserisce nello slot sotto la tastiera al posto della vecchia espansione da 512K, completa di clock in tempo reale e batteria tampone 450.000
Espansione da 2 MB esterna per A-500 o A-1000 .... 799.000
Hard disk GVP Impact 40 MB per A-500 .... 1.550.000
Hard card GVP 40 MB per A-2000 .... 1.480.000
Hard card GVP 100 MB per A-2000 .... 2.550.000
Velocizzatore 68030 GVP A-3001 .... da 1.440.000

SYNCHRO EXPRESS

Eccezionale novità per Amiga: è finalmente disponibile il primo **copiatore hardware** per i dischetti Amiga! Con una speciale interfaccia collegata a 2 disk drives (quello interno al computer ed uno esterno), effettua copie di sicurezza, perfettamente funzionanti, di **qualsiasi software protetto** in meno di 50 secondi, compresi gli "impossibili" come Dragon's Lair.
89.000

Dischi Fish di pubblico dominio aggiornati al n. 240

FATTER AGNUS 8372-A

Il nuovo chip che permette di usare 1 MB di Chip Ram nel vostro Amiga, disponibile ora in kit di montaggio per l'installazione in tutti i modelli B-2000, ed A-500 (con piastra madre rev. 4 o 5) con inserita l'espansione A-501 da 512K.
159.000

Prezzi IVA compresa

Viale Monte Nero 31
20135 Milano
Tel. (02) 55.18.04.84
(4 linee ric. aut.)
Fax (02) 55.18.81.05 (24 ore)
Negozio aperto al pubblico tutti i giorni dalle 10 alle 13 e dalle 15 alle 19.
Vendita per corrispondenza.
Sconti per quantità ai sigg. Rivenditori.

SUPERMENU PROGRAMMABILE

La possibilità di usare il mouse si scontra con la difficoltà di programmarlo in modo versatile. Ecco quindi un programma che intercetta e gestisce le sue coordinate

di Alessandro de Simone

Nei programmi professionali è sempre presente la possibilità di effettuare scelte mediante mouse.

Di solito si posiziona la freccetta su un rettangolo (in cui, magari, è conteuto un messaggio o un simbolo) e, cliccandovi, viene attivata la funzione corrispondente.

Un programma che svolga il compito descritto, implementato in qualsiasi linguaggio, deve, per forza di cose...

1 Disegnare i vari rettangoli (o "bottoni") che rappresenteranno le scelte da effettuare.
2 Individuare le coordinate del mouse.
3 Verificare se queste cadono all'interno di uno dei rettangoli visualizzati.
4 Attivare la procedura richiesta.

Ogni rettangolo che compare su video avrà coordinate **Xin**, **Yin** ed **Xfin**, **Yfin**; queste indicheranno, rispettivamente, le coordinate del vertice in alto a sinistra ed in basso a destra del rettangolo stesso.

Una routine in grado di determinare se il puntatore del mouse ("**M**", di coordinate **Xmou**, **Ymou**) "cade" all'interno del grenerico rettangolo (vedi **figura 1**) può essere, a livello simbolico, la seguente:

X = 0: Y = 0
IF Xmou >= Xin and Xmou <= Xfin THEN X = 1
IF Ymou >= Yin and Ymou <= Yfin THEN Y = 1
IF X = 1 AND Y = 1 THEN PRINT "Il mouse è all'interno": END
PRINT "Il mouse è all'esterno"

Se siamo in presenza di più rettangoli sarà necessario scrivere una subroutine del tipo descritto per ciascuno di essi.

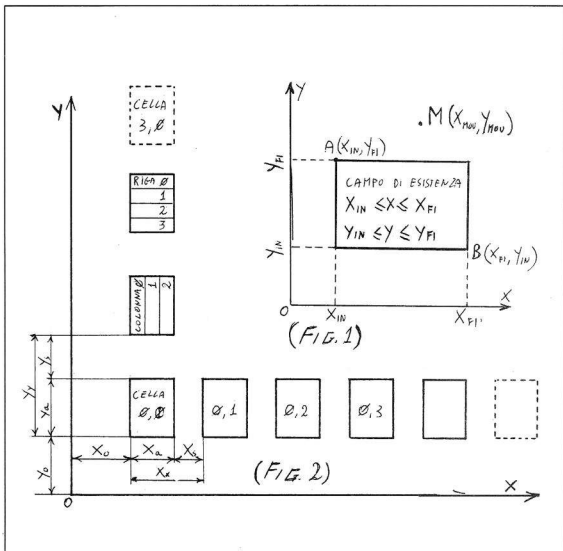
Si può facilmente immaginare che già in presenza di una diecina di rettangoli il programma si allunga enormemente ed i tempi di elaborazione possono diventare inaccettabili. Purtroppo questa è l'unica via da seguire nel caso in cui i vari bottoni siano numerosi, di dimensione diversa tra loro e sparsi ovunque nello schermo. Se, però, i rettangoli sono sistemati in modo lineare, sono eguali tra loro e disposti ordinatamente in una matrice di righe e colonne, il problema si affronta (e si risolve) con una certa facilità dal momento che è possibile ricorrere a semplici iterazioni.

Per chi inizia

Il programma di queste pagine è destinato a chi si è procurato da poco tempo un Amiga. La sua lunghezza, infatti, è sufficientemente breve(!) da invogliare il lettore alla sua digitazione. Questa deve essere effettuata prestando la massima attenzione ai valori numerici presenti al suo interno; con particolare cura devono esser trascritti gli argomenti delle istruzioni, eventualmente presenti, Peek Poke, Call.
Anche se non riuscite a ben comprendere il reale significato delle varie istruzioni, non scoraggiatevi: digitate il tutto e registrate su dischetto (comando **Save**) il programma pubblicato.
Il valore del programma è anche didattico. E' infatti possibile rendersi conto delle varie difficoltà che si incontrano nello scrivere un listato, benchè semplice come quello che compare in queste pagine. Figuratevi, quindi, la pazienza che è necessaria per realizzare listati pieni di sprite, schermate in alta risoluzione, musica ed animazioni varie! A volte può capitare che il programma presenti malfunzionamenti a causa di una non corretta trascrizione del listato. Nei casi più fortunati viene emessa una segnalazione di Syntax Error oppure di Illegal Quantity Error. In altri casi si rischia di bloccare il computer. C'è infine una terza eventualità: in caso di errore il programma gira e non segnala errori, ma l'elaborazione procede in maniera anomala ed erronea. Si consiglia ancora, pertanto, di digitare il programma con la massima attenzione e di registrarlo sempre, prima di impartire il comando Run.

Rettangoli

Il problema si limita(!) ad individuare una relazione matematica che consenta di stabilire se un certo punto, di coordinate generiche **X, Y** (che rappresenterà il mouse), appartenga all'area di un rettangolo, e a quale di essi.

Nella **figura 2** viene schematizzata una situazione reale nella quale sono presenti quattro righe e sei colonne; in corrispondenza di ciascun incrocio tra riga e colonna è presente un rettangolo. E' indispensabile definire le variabili che entrano in gioco:

X0: è la distanza dall'asse Y (più tardi corrisponderà al bordo dello schermo) dalla prima colonna.

Xa: è la larghezza (fissa) del rettangolo generico.

Xs: è la distanza (fissa) tra due rettangoli adiacenti.

Ne consegue che...

Xx = Xa + Xs

In modo analogo vengono definite le variabili **Y0, Ya, Ys, Yy** relative alle misure lungo l'asse Y.

Il numero di colonne e di righe viene banalmente indicato in **Rig** e **Col**.

Ragionando in modo adeguato è facile giungere alla conclusione che un punto **M(X, Y)** appartiene ad uno qualunque dei rattangoli se, e solo se, si verificano contemporaneamente le due condizioni seguenti:

```
X0 + (Rig * Xx) <=X <= X0 + ((Rig + 1) * Xx) - Xs

Y0 + (Col * Yy) <=Y <= Y0 + ((Col + 1) * Yy) - Ys
```

I segni di eguale (=) devono essere omessi se si desidera che il punto non appartenga al bordo del rettangolo.

Le due relazioni matematiche (la cui individuazione ha richiesto la collaborazione della prof. Giaccaglia, alla quale vanno i miei ringraziamenti) tengono conto del fatto che le aree delle singole celle rappresentano superfici non contigue.

In definitiva, individuate le coordinate del mouse, è necessario, mediante cicli

For... Next, esaminare, per ciascuna colonna e per ciascuna riga, se si verificano le condizioni indicate.

In pratica, però, spesso è sufficiente effettuare un solo controllo: se la coordinata X, ad esempio, è esterna all'intervallo dovuto, è inutile effettuare controlli anche lungo l'asse Y.

Di solito, infatti, definiamo priva di significato la pressione del tasto del mouse, se questa si verifica in un punto qualunque all'esterno dei rettangoli.

Il programma

Tra il dire e il fare, ovviamente, c'è una qualche difficoltà di implementazione, aggravata dal desiderio di inserire automatismi vari che consentano di rendere il programma decisamente versatile e di impiego universale.

Incominciamo, comunque, col descrivere le varie routine presenti nel listato.

Il programma inizia con la richiesta di vari parametri, deducibili ancora dalla figura 2.

Il primo **Input** si riferisce ad una non meglio identificata **matrice**, da misurare in **pixel**. Il programma, come vedremo, consente di inserire, all'interno di ciascun rettangolo, un messaggio. Questo, se è formato da caratteri standard Amiga (di matrice 8 x 8 pixel) potrebbero presentare fenomeni di disallineamento in una successiva fase di visualizzazione. Si consiglia, pertanto, di effettuare i primi esperimenti rispondendo con **8** alla prima domanda.

Il secondo e terzo Input si riferiscono alla dimensione del singolo rettangolo, da misurare in **righe** e **colonne** (di caratteri) ciascuna di grandezza precedentemente indicata in pixel.

Spieghiamoci meglio: se avete prima risposto con 8 alla domanda sui pixel ed ora assegnate 4 (altezza cella) e 6 (larghezza cella), ognuna di esse potrà, almeno in teoria, ospitare un messaggio di 24 (6x4) caratteri: la dimensione di ciascun rettangolo-cella, misurato in pixel, sarà invece di 32 (4x8) pixel di altezza e 48 (6x8) di larghezza. Tali dimesioni possono essere utili per eventuali sofisticazioni del programma.

Segue quindi la richiesta del numero di rettangoli da far apparire su video, misurati in righe e colonne di "reticolo".

In seguito, ad X0 ed Y0 verranno associate le distanze dal bordo della prima riga e della prima colonna, rispettivamente lungo l'asse X ed Y.

Alla domanda sui **Data**, rispondete, per ora, con un tasto qualsiasi diverso da "**S**"; ad una successiva prova, quando avrete meglio compreso il funzionamento del programma, risponderete "s".

Subito dopo vengono inizializzate le variabili ai valori che, in seguito, saranno correttamente elaborati.

Per non complicare le cose (ma anche per costringervi a ragionare almeno un po'...) la distanza tra righe e colonne è stata posta pari ad una riga (vedi **Ys = Mat; Xs = Mat**). Nulla vieta di distanziare le celle in modo differente (o di annullare la distanza stessa), ricorrendo ad altre variabili ed altrettanti Input.

A questo punto vengono tracciati, riga per riga, in vari rettangoli costituenti il "reticolo". Se avete esagerato assegnado valori poco credibili (un milione di colonne, matrice di 1x1 pixel e simili preziosismi) è probabile che venga segnalata qualche irregolarità. Vi consigliamo, almeno all'inizio, di rispondere ai vari Input come indicato in tabella. Poi sbizzarritevi come vi pare e piace.

Da notare che AmigaBasic considera lo schermo con l'origine degli assi in alto a sinistra, l'asse Y positivo verso il basso e quello X positivo verso destra. Alcune difficoltà possono presentarsi, quindi, comparando la figura 2 con quanto appare su video.

Appena date **Run**, e rispondete come indicato in tabella, un meraviglioso reticolo di 32 (4x8) rettangoli apparirà sul video. Il primo di questi, inoltre, presenterà il bordo lampeggiante. In effetti non si tratta di un lampeggio vero e proprio, ma di un banale *colora - e - cancella - bordo - del - rettangolo.*

I singoli rettangoli appaiono grazie al doppio ciclo **For... Next (J, I)** relativo a righe e colonne.

Dal momento che abbiamo risposto con "N" alla richiesta della lettura dei Data, l'elaborazione salta ad eseguire la routine **SoloReticolo**. A partire da questa inizia la gestione del reticolo vera e propria.

Il mouse

Anzitutto **Col** e **Rig** vengono poste a zero, riferendosi al rettangolo di default.

La variabile **a** viene incaricata di individuare l'eventuale pressione del tasto sinistro del mouse. Se questo non viene premuto (**a = 0**) la routine **Loop** viene ripetuta all'infinito. Al suo interno, però, è presente la routinetta che fa lampeggiare il bordo dell'ultimo rettangolo selezionato (o quello di default, se ci troviamo subito dopo il Run).

Se, invece, **a** è diversa da zero, vuol dire che il tasto è stato premuto; in questo caso ad **X** ed **Y** vengono associate le coordinate assolute del mouse, pronte per esser trattate dalla routine "cuore" del sistema.

Prima di far questo, però, le variabili **Riga** e **Colonna** vengono impostate al valore negativo **-1**.

I due cicli For... Next (i, j) successivi verificano se le coordinate del mouse "cadono" all'interno di uno dei rettangoli e, in caso affermativo, associano a Riga e Colonna il loro corrispondente posizionamento. In caso contrario, una delle due variabili Riga e Colonna (o entrambe) non vengono modificate: la presenza del valore negativo precedentemente impostato (anche in una sola delle due variabili) permetterà di riconoscere come priva di significato la posizione del mouse. Si noti che i due cicli For... Next **non** sono nidificati, e non devono esserlo. In caso contrario, infatti, si allungano i tempi di elaborazione e si giunge alla stessa conclusione (appartiene ad un rettangolo **sì**, appartiene **no**). Se abbiamo clickato in uno dei rettangoli, tale

Matrice? **8**
Altezza cella? **3**
Larghezza cella? **5**
Righe? **4**
Colonne? **8**
Interd. orizz.? **3**
Interd. vert.? **2**
Leggo Data? **n**

Tabella
Si consiglia di rispondere come indicato, almeno per verificare il corretto funzionamento del programma.

```
REM Programma AmigaBasic per la generazione di un "reticolo" gestibile
REM da mouse (con successivo riconoscimento della cella clickata)
' by A. de Simone 1990

INPUT "matrice X * Y pixel (consigliabile 8)"; mat
INPUT "altezza singola cella (in RIGHE 1 - 4)"; h
INPUT "larghezza singola cella (in COLONNE 1 - 9)"; l
INPUT "n. righe reticolo (1 - 4)"; r: r = r - 1
INPUT "n. colonne reticolo (1 - 9)"; c : c = c - 1
INPUT "distanza orizzontale da bordo video (in righe)"; x0
INPUT "distanza verticale da bordo video (in colonne)"; y0
INPUT "leggo DATA presenti in programma (s/n)"; ld$
IF UCASE$(ld$) = "S" THEN ld = 1
CLS: ' Inizializzazione variabili e cancellazione schermo
y0 = y0 * mat: ya = h * mat: ys = mat: yy = ya + ys: ny = r
x0 = x0 * mat: xa = l * mat: xs = mat: xx = xs + xa: nx = c
' Generazione dei rettangoli costituenti il reticolo
FOR j=0 TO ny
FOR i=0 TO nx
LINE (x0 + xx * i, y0 + j * yy) - (x0 + xx * (i + 1) - xs, y0 + yy * (j + 1) - ys), 3, bf
NEXT i, j
IF ld =0 THEN SoloReticolo: ' Salta la lettura dei DATA
(Vedi prima, INPUT LD$)
RESTORE Messaggi: fineRead = 0: ' Legge i dati dai DATA
FOR i=0 TO nx
FOR j=0 TO ny: IF fineRead = 1 THEN a$ = "prova": GOTO NoRead
READ a$: IF a$="*" THEN fineRead = 1
' Trucchetto per non scrivere troppi DATA...
NoRead: ' nel caso in cui i DATA fossero insufficienti, inserisce "PROVA"
lu = 2: IF xa < 3 * mat THEN lu = 1: IF xa = mat THEN lu = 0
Riga = 1 + (ya / mat) / 2 + (yy / mat) * j + y0 / mat
Colonna = 1 + lu + (xa > 2 * mat) + (xx / mat) * i + x0 / mat
LOCATE Riga, Colonna: ' posizionamento all'interno della singola cella
PRINT LEFT$ (a$,- lu + xa / mat): ' trascrizione (di parte) del messaggio
NEXT j, i
SoloReticolo:
col = 0: rig = 0: ' default = cella n. 0, 0 (riga, colonna)
Loop:
a = MOUSE (0): i = rig: j = col
' formula magica per individuare l'ultima cella selezion. e farla "lampeggiare"
LINE (x0 + xx * i, y0 + j * yy) - (x0 + xx * (i + 1) -xs, y0 + yy * (j + 1) -ys), 2, b
LINE (x0 + xx * i, y0 + j * yy) -(x0 + xx * (i + 1) -xs, y0 + yy * (j + 1) -ys), 3, b
IF a = 0 THEN Loop: ' lampeggia finche' non si muove il mouse
x = MOUSE(1): y = MOUSE(2): ' coordinate mouse al momento del click
Riga = -1: Colonna = -1: ' inizializzazione prima di elaborazione dati mouse
FOR i = 0 TO nx: ' ...verifica se la coordinata X e' compatibile con impostazioni
IF x >= x0 + i * xx AND x <= x0 + (i + 1) * xx -xs THEN Riga = i
NEXT i: IF Riga = -1 THEN GOTO FuoriAreaX: '...altrimenti e' inutile controllare
FOR j = 0 TO ny: ' ...verifica se la coordinata Y e' compatibile...
IF y >= y0 + j * yy AND y <= y0 + (j + 1) * yy -ys THEN Colonna = j
NEXT j
FuoriAreaX: ' se le variabili RIGA e COLONNA sono NON negative,
' allora abbiamo cliccato su cella "esistente"
' attenzione: R = colonna, C = riga (e' un piccolo bug scoperto
' all'ultimo momento; perdonatemi...)
IF Riga >=0 AND Colonna >=0 THEN LOCATE 1, 1: PRINT "col." Riga; "rig."
Colonna: rig = Riga: col = Colonna ELSE BEEP
```

condizione verrà evidenziata (vedi **Locate... Print**) dalla coppia dei valori che individuano la cella.

A questo punto è possibile inserire il "dirottamento" desiderato che, nel programma di queste pagine, non solo fa comparire solo un banale messaggio, ma è addirittura limitato a due casi particolari: l'indicazione della prima o dell'ultima cella visualizzabile.

Le subroutine

E' infatti necessario, a questo punto (vedi **EseguiCompiti:**), inserire tutte le subroutine che devono essere eseguite a seconda del valore di **Riga** e **Colonna** determinati. Tale procedura è difficilmente automatizzabile perchè bisognerebbe avere a disposizione tante subroutines quante sono le celle impostabili. Nel listato pubblicato ne compaiono solo due (**PrimoComando** e **UltimoComando**) che sono le uniche che possono essere stabilite a priori dal momento che ci sarà sempre la prima e... l'ultima cella di un qualsivoglia numero di righe e colonne.

Il listato, comunque, non per questo perde la sua validità e versatilità: l'attento lettore troverà certamente il modo di estrarre una (o più) procedure ivi inserite per realizzare facilmente menu gestibili con il mouse. E' sufficiente ricordare che una qualsiasi subroutine deve iniziare con...

SUB NomeRoutine STATIC

...terminare con...

END SUB

...ed essere "invocata" con...

CALL NomeRoutine

Maggiori dettagli sul corretto uso di una Subroutine, e sulle sue variabili, possono essere rintracciate sul manuale AmigaBasic.

Messaggi incorporati

All'interno di ciascun rettangolo è possibile trascrivere un messaggio;

```
' inserire le varie chiamate alle subroutines a seconda del valore
' di RIGA e COLONNA

EseguiCompiti:
IF Riga = 0 AND Colonna = 0 THEN CALL PrimoComando
' inserire, uno per uno, i vari compiti da svolgere a seconda del valore di Riga e Colonna
IF Riga = c AND Colonna = r THEN CALL UltimoComando: END
x1 = x: y1 = y
Controllo: ' impedisce di ritornare al ciclo principale se si indugia sul p
a = MOUSE(0): x = MOUSE(1): y = MOUSE(2)
IF x = x1 AND y = y1 THEN Controllo:
GOTO Loop: ' si ricomincia
Messaggi: ' il numero di DATA deve essere coerente con matrice di celle. In caso contrario l'asterisco in fondo provvede
ad evitare errori del tipo "OUT OF DATA"

DATA uno, due, tre, quattro, cinque, sei, sette, otto, nove, dieci
DATA alfa, beta, gamma, delta, epsilon, zeta, eta, teta
DATA uomo, donna, misto, *
'
SUB PrimoComando STATIC
PRINT "Hai clickato nella cella 0, 0. Contento?"

END SUB
'
SUB UltimoComando STATIC
PRINT "Hai clickato sull'ultima cella. Io ho finito"

END SUB
```

questo può essere prelevato da un gruppo di **Data** (come nel programma pubblicato) oppure appartenere ad un file precedentemente memorizzato.

La sua corretta trascrizione, all'interno di un rettangolo, si scontra sia con le dimensioni del rettangolo stesso, sia con la lunghezza della stringa da trascrivere.

Il secondo problema si risolve facilmente troncando brutalmente il messaggio in modo che risulti non più lungo della larghezza della cella (del resto, non si può agire diversamente).

La soluzione del primo problema, invece, dipende dai... gusti personali.

Può far piacere, infatti, inserire il messaggio al centro di una cella; se la cella-tipo è alta, ad esempio, 5 (numero **dispari**) righe, la stringa viene visualizzata sulla terza riga (cioè perfettamente centrata). Tale "perfezione" non può riscontrarsi nel caso di altezza pari (2, 4 , 6 ecc. righe).

In questo caso, infatti, il programma pubblicato presenta inevitabilmente un "anestetismo", se con questo termine vogliamo indicare l'assenza di simmetria di sapore vanvitelliano (ma chi l'ha detto che una figura asimmetrica non è bella?).Chi lo desidera, ovviamente, può tentare di far apparire il messaggio-stringa sempre ai "piedi" della cella, o sempre in alto.

La routine che gestisce il posizionamento della frase è quella che parte da **Restore Messaggi** e giunge fino a **SoloReticolo**.

E' anche presente, al suo interno, la routine che consente di posizionare il messaggio nella cella voluta.

Dal momento che, con il programma pubblicato, è possibile far apparire decine di righe e colonne, il numero di Data presenti dovrebbe essere in misura corrispondente.

Per evitare di allungare il brodo sono state inserite solo tre righe di Data e l'ultimo dato è un asterisco(*).

I dati presenti, quindi, vengono inseriti uno alla volta se la matrice è di modeste dimensioni. In caso contrario, al momento dell'intercettazione dell'asterisco, i rettangoli in sovrannumero verranno riempiti tutti della stringa "**Prova**" (o di parte di essa, in dipendenza della larghezza della cella-tipo).

Un ultimo particolare è giusto che venga notato, se non altro per sottolineare il desiderio di inserire accorgimenti che possono rivelarsi utili in casi particolari.

Nel caso in cui le celle visualizzate siano piuttosto numerose, potrebbe risultare difficile ricordare quale di esse sia stata selezionata per ultima (pensate ad un gioco di scacchiera, che può benissimo esser implementato sviluppando l'idea...).

Se il mouse rimane posizionato su di essa, infatti, non vi sono problemi sulla sua individuazione.

Se, però, dopo aver clickato spostiamo il mouse in una posizione priva di significato, la prima cella (0, 0) verrebbe individuata come default.

Per evitare questo inconveniente, la cella selezionata per ultima continuerà a lampeggiare grazie alla re-impostazione continua dei valori di default.

La routine **Loop**, infatti, considera come default i valori **i** e **j**, a loro volta "inizializzati" a **Rig** e **Col** tutte le volte che (vedi routine "**FuoriArea**") viene premuto il tasto del mouse all'interno di una cella "legale".

IL QUIZZER

Pensate di essere esperti di Amiga? Allora provate a rispondere alle seguenti domande. Confrontando le risposte forse troverete qualcosa di curioso ed interessante da imparare...

a cura di Luigi Callegari

A Quando si usa il comando *Setclock LOAD* da file di startup o dallo Shell si ottiene *Internal Clock not functioning*, a causa di qualche programma impazzito, e l'orario risulta errato e non assegnabile nonostante l'orologio sia montato. Come fare per rimettere tutto a posto?

a) Uso Preferences per rimettere l'ora
b) Uso il comando SetClock della V1.2
c) Lascio spento Amiga per un poco

B Devo copiare alcuni files, e non tutti, da una directory ad un'altra usando lo Shell standard (non ARP). Dopo avere effettuato un *CD* nella directory dei files da copiare, qual è il modo più veloce (o l'unico) per ottenere la copia?

a) Uso *Copy* per i vari nomi
b) Uso l'istruzione *Rename*
c) Uso *Copy* e l'operatore | sulla linea.

C Il primo virus storicamente posto in circolazione per Amiga è stato:

a) Byte Killer (o Byte Bimbo)
b) SCA
c) Raffreddurum

D Quanti sono approssimativamente i nodi della rete *Fido BBS* in Italia attualmente?

a) Un centinaio
b) Due centinaia
c) Una ventina e mezza

E I redattori / collaboratori di CCC, i cui nomi appaiono sempre nella prima pagina della rivista, sono approssimativamente:

a) Una trentina
b) Una ventina
c) Dei morti di fame

F Il famoso linguaggio interprete AmigaBasic è stato prodotto:

a) Dalla mitica Microsoft
b) Dalla giovane Commodore Amiga
c) Dalla inglese Metacomco

G Il compilatore Lattice C V5 per Amiga è famoso principalmente perchè:

a) E' quasi esente da bug
b) Originariamente era pieno di bug
c) Costa troppo

H Il nome BASIC significa effettivamente:

a) Beginner's Allpurpose Symbolic Interactive Code
b) Basta Andare Sempre Indietro Computerizzando
c) Beginner's Allpurpose Sintactical Interactive Code

I La differenza principale tra Aztec C e Lattice C versioni 5 è che:

a) Aztec genera direttamente codice assembly
b) Lattice è compatibile ANSI, Aztec no
c) Il compilatore di Aztec scrive più parolacce

J Quale delle seguenti tre sequenze comprende una parola che non c'entra con le altre:

a) For, Print, Dim, Option, Let, Open
b) for, sprintf, int, long double, fopen
c) move, lea, btst, clr, jr, rts, subq

K Avendo un gruppo di files del Ram Disk, ad esempio test1, test2, test30 e test40, voglio usare lo Shell per cancellarli tutti in una volta. Però all'interno del *RAM DISK* ho anche una directory

chiamata test5 con dei files importanti al suo interno.

La cosa più saggia è:

a) Usare **Delete test#?** tranquillamente

b) Devo usare prima *Rename* per rinominare la directory

c) Lasciare perdere, risparimando sui polpastrelli

L La sigla I.F.F. che spesso compare negli articoli su Amiga significa:

a) International File Format

b) Inter Football Fans

c) Interchange File Format

M Bug significa "errore" in gergo, ma letteralmente invece:

a) Pidocchio

b) Pulce

c) Un fan del complesso The Bug's

N Una word di memoria può contenere effettivamente:

a) Sedici bit

b) Trentadue bit

c) Un numero di Bit predeterminato.

O Un programma compilatore, in fin dei conti, si può assimilare a:

a) Un programma che trasforma un testo ASCII in assembly.

b) Un programma che esegue un testo ASCII.

c) Un testo ASCII che esegue un programma (ma solo da CLI).

P In linguaggio informatico la parola *Flag* ha il significato di:

a) Procedura fallita ("Il computer è andato in flag").

b) Un segnalatore di evento o di stato logico.

c) Un tipo di file ad accesso casuale.

Q I files IFF, che Amiga è in grado di gestire, possono essere:

a) Grafici, Sonori e Testuali

b) Grafici e Sonori

c) Dipende dalle implementazioni

R La ROM di Amiga nella V2.0 del sistema è di 512K; nel mitico VIC-20 era invece di:

a) 1024K

b) 16K

c) 20K

S Il linguaggio di programmazione C è nato:

a) Nei Bell Telephone Laboratories

b) Presso la AT & T

c) Presso la Commodore Italia

T La differenza tra monitor 1084 e 1084S per Amiga consiste:

a) Nei fosfori ad alta persisenza.

b) Nella previsione di due casse per la stereofonia.

c) Nel prezzo.

U L'istruzione DIVS del processore 68000 richiede per essere eseguita:

a) 122 cicli di clock

b) 108 cicli di clock

c) Dipende dal clock settato con Preferences.

Risposte

A b. Con l'opzione *Reset*, oppure usando *Setclock* della V1.3.2.
Le altre due operazioni non servono a nulla quando il registro dell'orologio è corrotto.

B c. Ad esempio...
COPY sys:C/(Dir|List|Copy) TO RAM:
...esegue la copia in RAM DISK dei tre files tra parentesi leggendoli da SYS:C.

C b. Seguito comunque, a brevissima distanza, dal molto più cattivo Byte Killer.

D a. Ma il calcolo è abbastanza difficile, in quanto ne aprono e ne chiudono in continuazione.

E a. Se avete invece risposto (c) forse non leggete mai le gerenze, ma probabilmente conoscete alcuni di noi.

F a. La Commodore Amiga aveva originariamente fornito ABasic della Metacomco Plc (softhouse inglese, colpevole anche di AmigaDOS V1.0/1.3), ma in Italia non lo abbiamo mai visto (per fortuna) nelle macchine importate dalla Commodore Italia.

G b. Comunque, nonostante varie upgrade, ne ha ancora di belli. Anche la risposta (c) non è apparentemente troppo sbagliata secondo la mentalità degli utenti Commodore. Si pensi comunque che il Microsoft C per MS/DOS costa più del doppio.

H a. Ovvero: Codice interattivo simbolico di applicazione generale per principianti.

I a. Le versioni 5.0a di Aztec e 5.04 di Lattice sono ANSI compatibili.

J c. L'istruzione **jr** non esiste nel set di istruzioni assembly del 68000, mentre le altre sono gruppi corretti di istruzioni BASIC e C.

K a. Infatti, se non specifico il parametro ALL, la directory non verrà cancellata.
Se avete risposto (c) non diventerete mai dei grandi programmatori.

L c. Infatti è stato sviluppato dalla Electronic Arts per consentire lo scambio di files tra vari computers.

M b, da cui gli orribili neologismi "debacare" , "debuggare" e via orripilando.

N a, mentre 32 bit rappresentano una longword, e 8 un byte.

O a, mentre (b) potrebbe essere la definizione di un Interpete di linguaggio.

P b, mentre (c) non c'entra proprio nulla.

Q Sebbene molti possano avere risposto **a**, in effetti è **c**. Infatti si tratta di uno standard ampliabile, previ contatti con la Electronic Arts oppure con i CATS (Commodore AMiga Technical Support) americani.

R c, mentre 16K erano del Sinclair Spectrum e del Sinclair ZX 81.

S a, anche se non si tratta di un linguaggio per la gestione dei telefoni.

T b, ma anche il vecchio 1081 veniva venduto in modelli stereofonici ed in modelli monofonici allo stesso prezzo.

U a, mentre (b) è per l'istruzione DIVU. Se avete risposto giusto vuol dire che siete dei progettisti della Motorola, o semplicemente fortunati: fareste bene a giocare più schedine durante il campionato di calcio perchè farete sicuramente 13.

AMIGA, CONOSCERLA PRIMA DI COMPRARLA

Come l'evoluzione del mercato faceva già prevedere, è in continuo crescendo la fascia di utenti che abbandonano i "piccoli" computer di marca Commodore per accostarsi al ben più allettante mondo di Amiga.

Molti di questi, vecchi lettori e non, ci bersagliano di lettere per esprimere i loro dubbi o richiedendoci consigli sull'imminente acquisto, riproponendo temi che più di una volta sono stati affrontati nelle pagine della rubrica Postamiga.

Non potendo occupare altre pagine per ribadire argomenti già approfonditi, ecco quindi un sintetico "decalogo" dedicato a tutti i quasi o neo-amighi, tra i quali vanno annoverati Salvatore Incardona, Sandro Francini, Marco Alzetta, Niccolò Badoglio, Marco Tiramani, ed una marea di anonimi dalla firma illeggibile (o dimenticata).

Come ovvio, alcuni pareri sono del tutto personali, e vanno interpreti come consigli fraterni, più che verità assolute.

Drive

A meno di non voler tenere in soffitta, magari per ricordo, tutto l'armamentario legato a C/64, C/128, eccetera, meglio **togliersi subito dalla testa** di utilizzare i disk drive associati a questi computer con Amiga.

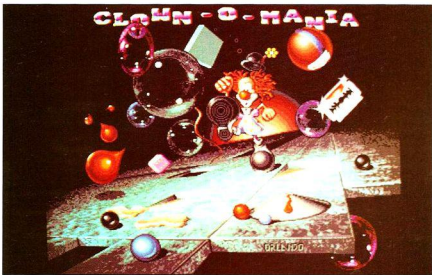
Molti hanno leggiucchiato qua e là di "cose" come il C/64 Emulator, ma è opportuno sgombrare il campo dagli equivoci: Amiga si adopera in un altro modo! Simili collegamenti sono possibili solo in via temporanea, o giusto come curiosità.

Il che significa: se, per esempio, si possiede una grossa mole di testi elaborati col buon vecchio Easy Script e li si vuole riutilizzare in futuro, allora potrebbe risultare utile collegare un drive 1541 (e compatibili) all'Amiga tramite l'emulatore, trasferire i dati, dopodichè... gettare alle ortiche il drive.

E non a caso si è parlato di testi: i programmi per Amiga possono essere caricati solo da dischi per Amiga, inseriti in Drive per Amiga! Quando avrete compiuto il "grande passo", vedrete come, se proprio si vuole emulare qualcosa (il perchè, poi, sfugge alla nostra comprensione) Amiga può rivolgersi a sistemi ben più evoluti di un "misero" C64/128.

Stampante

Amiga supporta pressochè tutti i tipi di stampante, purchè dotati di interfaccia parallela (standard **Centronics**). Modelli di marche "strane" possono andar bene nella maggior parte dei casi: accertarsi, se già si possiede qualcosa del genere, che la periferica possa operare in **emulazione Epson**, oppure emulazione **Ibm**. Le stampanti seriali per C64/128 (per intenderci, quelle collegabili al computer con il tipico spinotto "rotondo"), sono assolutamente da scartare.

Chiaro che, dovendone acquistare una nuova, il discorso si fa più semplice, visto che la stessa Commodore produce una gamma di modelli perfettamente adattabili ad Amiga. Chi volesse tenere comunque l'otto bit, in questo caso può rivolgere la sua scelta ad un modello dotato sia di interfaccia seriale Commodore che parallela Centronics.

Modelli Amiga

Molti lettori sono preoccupati dal numero di sigle che caratterizzano la gamma Amiga: **500, 2000, 3000**.

Preoccupati, soprattutto, dalla possibilità di dover ancora una volta "rincorrere" il mercato, trovandosi di fronte ad incompatibilità tra macchiene diverse.

Ebbene, con Amiga è pressochè impossibile non scontrarsi prima o poi con qualche incompatibilità.

La cosa, però, non deve preoccupare più di tanto.

Il fatto è che questo computer è rivolto ad una fascia molto differenziata di utenza: dallo smanettone casalingo tutto games e joystick, al professionista che necessita di una work station grafica. Quest'ultimo non acquisterà certo un A-500, ma d'altra parte è alquanto improbabile che un hobbista spenda **7 milioni** (e passa) per un Amiga 3000.

Chiaro che, se per esempio viene progettato un software professionale che necessita di 2 Megabyte di memoria, coprocessore matematico, ed altra robetta simile, questo non potrà mai funzionare su un A-500 in versione base.

Di contro, almeno finora, i migliori prodotti di word processing, data base, eccetera, possono tranquillamente girare su tutte le macchine, in qualche caso (per A-500) con la semplice aggiunta di una espansione di memoria.

Si pensi che c'è ancora chi usa l'Amiga 1000, ormai fuori produzione da un bel po' di tempo, preferendolo addirittura ai modelli successivi...

La scelta, in definitiva, è legata alla cifra che si intende spendere, nonchè all'uso che si prevede per il nuovo gioiello.

C'è da dire che, anche se con qualche ritardo rispetto alla immissione sul mercato delle novità, di solito è sempre possibile aggiornare la propria macchina con l'aggiunta di qualche scheda o una sostituzione di componenti, senza dover ricomprare tutto. Amiga, insomma, è un sistema "aperto", non rigido come i vecchi 64, 128 e compagnia: caratteristica, questa, che la rende quantomeno più al riparo da improvvisi crolli.

Senza dimenticare, però, che per l'informatica un lasso di dieci anni è una eternità, in grado di rendere obsoleta qualunque rivoluzione dell'ultima ora.

Prevedere tutto

Un Amiga 500 in configurazione base può già dare notevoli soddisfazioni, soprattutto se si era abituati a livelli inferiori, ma è opportuno prevedere, in questo caso, ulteriori acquisti a breve termine, se non proprio contemporanei. Intanto, l'orientamento generale si va spostando verso una disponibilità di memoria **non inferiore al Megabyte**, tanto che lo stesso A-500 verrà probabilmente configurato di base con tale quantità di Ram.

All'acquisto del modello più economico, quindi, è consigliabile associare al più presto una espansione di memoria (dal prezzo, in fondo, abbastanza limitato).

Un solo drive, poi, ad Amiga va veramente stretto, considerata la sua peculiare gestione dei files.

Ergo, **secondo drive pressochè obbligatorio**; anche in questo caso, comunque, non si tratta di spese astronomiche.

Altro elemento da non sottovalutare è il video. Amiga, ormai è risaputo, è regina incontrastata nel settore della grafica.

Teoricamente può essere collegata anche ad un normalissimo apparecchio TV, ma perdendo quelle caratteristiche che in fondo la rendono così appetibile.

Un monitor, possibilmente dotato di audio stereo, è in pratica l'unico vero supporto hardware che possa garantire certe prestazioni.

In ogni caso, anche se non fosse possibile il "tutto e subito", si può star certi che Amiga farà presto dimenticare qualunque nostalgia per altre tastiere "appese al chiodo".

QUANTO COSTA IL TUO COMMODORE

Amiga 2000 - L. 2.715.000

Microprocessore Motorola MC68000 - Clock 7.16MHz - Kickstart ROM - Memoria RAM: 1 MByte - 3 chip custom per DMA, Video, Audio, I/O - 5 Slot di Espansione Amiga Bus 100 pin AutoconfigTM - 1 Slot di Espansione 86 pin per Schede Coprocessore - 2 Slot di Espansione compatibili AT/XT - 2 Slot di Espansione compatibili XT - 2 Slot di Espansione Video - 1 Floppy Disk Drive da 3 1/2", 880 KBytes - Porta seriale RS232C - Sistema Operativo single-user, multitasking AmigaDOS - Compatibilità MS-DOS XT/AT disponibile con schede interne Janus (A2088 - A2286) - Monitor escluso

Amiga 500 - L. 995.000

Microprocessore Motorola MC68000 - Clock 7.16 MHz - Kickstart ROM - Memoria RAM: 512 KBytes - 3 Chip custom per DMA, Video, Audio, I/O - 1 Floppy Disk Driver da 3 1/2", 880 KBytes - Porta seriale RS232C - Porta parallela Centronics

Videomaster 2995 - L. 1.200.000

Desk Top Video - Sistema per elaborazini video semiprofessionale composto da genlock, digitalizzatore e alloggiamento per 3 drive A2010 - Ingressi videocomposito (2), RGB - Uscite Videocomposito, RF, RGB + sync -

Floppy Disk Driver A 1010 - L. 335.000

Floppy Disk Driver - Drive esterno da 3 1/2" - Capacità 880 KBytes - Collegabile a tutti i modelli della linea Amiga, alla scheda A2088 e al PC1

Floppy Disk Drive A 2010 - L. 280.000

Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" - Capacità 880 KBytes - Collegabile ad Amiga 2000

Hard Disk A 590 - L. 1.750.000

Hard Disk+Controller+RAM - Scheda Controller - Hard Disk da 3 1/2" 20 MBytes - 2 MBytes "fast" RAM - Collegabile all'Amiga 500

Scheda Janus A 2088 + A 2020 - L. 1.050.000

Scheda Janus XT+Floppy Disk Drive da 5 1/4", 360 KBytes - Scheda Bridgeboard per compatibilità MS-DOS (XT) in Amiga 2000 - Microprocessore Intel 8088 - Coprocessore matematico opzionale Intel 8087

A2286+A2020 - L. 1.985.000

Scheda Janus AT+Floppy Disk Drive da 5 1/4", 1.2 MBytes - Scheda Bridgeboard per compatibilità MS-DOS (AT) in Amiga 2000 - Microprocessore Intel 80287 - Clock 8 MHz - RAM: 1 MBytes on-board - Floppy Disk Controller on-board - Floppy Disk Driver disegnato per l'installazione all'interno dell'Amiga 2000 -

Scheda A2620 - L. 2.700.000

Scheda Processore Alternativo 32 bit - Scheda per 68020 e Unix - Microprocessore Motorola MC68020 - Coprocessore matematico Motorola MC68881 (opzionale MC68882)

Scheda A Unix - L. 3.250.000

Sistema Operativo AT&T Unix System V Release 3 - Per Amiga 2000 con scheda A2620 e Hard Disk 100 MBytes

Hard Disk A2092+PC5060 - L. 1.020.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+2092 - L. 1.240.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+A2094 - L. 1.900.000

Stesse caratteristiche del kit A2092 ma con disco da 40 MBytes

Espansione di memoria A2058 - L. 1.149.000

Espansione di memoria - Scheda di espansione per Amiga 2000 - Fornita con 2 MBytes "fast" RAM, espandibile a 4 o 8 MBytes

Scheda Video A2060 - L. 165.000

Modulatore video - Scheda modulatore video interna per Amiga 2000 - Uscite colore e monocromatica - Si inserisce nello slot video dell'Amiga 2000

Genlock Card A2301 - L. 420.000

Genlock - Scheda Genlock semiprofessionale per Amiga 2000 - Permette di miscelare immagini provenienti da una sorgente esterna con immagini provenienti dal computer

Professional Video Adapter Card A2351 - L. 1.500.000

Professional Video Adapter - Scheda Video Professionale per Amiga 2000 (B) - Genlock qualità Broadcast - Frame Grabber - Digitalizzatore - Include software di controllo per la gestione interattiva
(Disponibile da maggio '89)

A501 - L. 300.000

Espansione di memoria - Cartuccia di espansione di memoria da 512 KBytes per A500

A520 - L. 45.000

Modulatore RF - Modulatore esterno A500 - Permette di connettere qualsiasi televisore B/N o colori ad Amiga 500

A Scart - L. 27.000
Cavo di collegamento A500/A2000 con connettore per televisione SCART

Monitor a colori 1084 - L. 595.000
Monitor a colori ad alta risoluzione - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor a colori 2080 - L. 770.000
Monitor a colori ad alta risoluzione e lunga persistenza - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Frequenza di raster 50 Hz - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor Monocromatico A2024 - L. 1.235.000
Monitor monocromatico a fosfori "bianco-carta" - Turbo 14" antiriflesso - (Disponibile da marzo '89)

PC60/40 - L. 7.812.000
Microprocessore Intel 80386 - Coprocessore matematico opzionale intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 funzioni - Sistema Operativo MS-DOS 3 2.1 - Interprete GW-Basic

PC60/40C - L. 8.127.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 60/80 - L. 10.450.000
Microprocessore Intel 80386 - Coprocessore opzionale Intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Memoria RAM: 2.5 MBytes - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Floppy Disk Drive opzionale da 3 1/2", 1.44 MBytes - 1 Hard Disk da 80 MBytes - 2 Porte parallele Centronics - Mouse video EGA (compatibile MDA - Hercules - CGA). Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Ambiente Operativo Microsoft Windows/386 - Interprete GW-Basic

PC60/80C - L. 10.700.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC40/20 - L. 4.100.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/20C - L. 4.350.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 40/40 - L. 5.285.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/40C - L. 5.535.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

1352 - L. 78.000
Mouse - Collegabile con Microsoft Bus Mouse - Collegabile direttamente a PC1, PC10/20 - III, PC40 - III

PC910 - L. 355.000
Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" per PC10/20-I-II-III - Capacità 360 o 720 KBytes selezionabile tramite "config. sys" - Corredo di telaio di supporto per l'installazione in un alloggiamento per un drive da 5 1/4" - Interfaccia identica ai modelli da 5 1/4"

PC1 - L. 995.000
Microprocessore Intel 8088 - 1 Floppy Disk Drive da 5 1/4" - Porta seriale RS232C - Porta parallela Centronics - - Monitor monocromatico 12" - Tastiera 84 tasti - Sistema Operativo MS-DOS 3.2 - Interprete GW-Basic

PCEXP1 - L. 640.000
PC Expansion Box - Box esterno di espansione per PC 1 - Alimentatore aggiuntivo incluso - Contiene 3 Slot di Espansione compatibili Ibm XT - Alloggiamento per Hard Disk da 5 1/4" - Si posiziona sotto il corpo del PC1 e viene collegato tramite degli appositi connettori

PC10-III - L. 1.360.000
Microprocessore Intel 8088 Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - Memoria RAM: 640 KBytes - 2 Floppy Disk Drive da 5 1/4", 360 KBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC10-IIIC - L. 1.675.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC20-III - L. 2.095.000
Microprocessore Intel 8088 - Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - 1/4", 360 KBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse) - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC20-IIIC - L. 2.410.000

Stessa configurazione ma con monitor 14" a colori mod. 1084

Nuovo C64 - L. 325.000

Nuovo Personal Computer CPU 64 KBytes RAM - Vastissima biblioteca software disponibile - Porta seriale Commodore - Porta registratore per cassette - Porta parallela programmabile -

C128D - L. 895.000

Personal Computer CPU 128 KBytes RAM espandibile a 512 KBytes - ROM 48 KBytes - Basic 7.0 - Tastiera separata - Funzionante in modo 128,64 o CP/M 3.0 - Include floppy disk drive da 340 KBytes

Floppy Disk Drive 1541 II - L. 365.000

Floppy Disk Drive - Floppy Disk Drive da 5 1/4" singola faccia - Capacità 170 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

Floppy Disk Dirve 1581 - L. 420.000

Floppy Disk Drive da 3 1/2" doppia faccia - Capacità 800 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

1530 - L. 55.000

Registratore a cassette per C64, C128, C128D

Accessori per C64 - 128D

1700 - Espansione di memoria - Cartuccia di espansione di memoria a 128 KBytes per C128 - **L. 170.000**

1750 - Espansione di memoria - Cartuccia di espansione di memoria 512 KBytes per C128 - **L. 245.000**

1764 - Espansione di memoria - Cartuccia di espansione di memoria a 256 KBytes per C64 - Fornita di alimentatore surdimensionato - **L. 198.000**

16499 - Adattatore Telematico Omologato - Collegabile al C64 - Permette il collegamento a Videotel, P.G.E. e banche dati **L. 149.000**

1399 - Joystick - Joystick a microswitch con autofire - **L. 29.000**

1351 - Mouse - Mouse per C64, C128, C128D - **L. 72.000**

Monitor Monocromatico 1402 - L. 280.000

Monitor monocromatico a fosfori "bianco-carta" - Turbo 12" antiriflesso - Ingresso TTL - Compatibile con tutta la gamma PC

Monitor Monocromatico 1404 - L. 365.000

Monitor monocromatico a fosfori ambra - Turbo 14" antiriflesso a schermo piatto - Ingresso TTL - Compatibile con tutta la gamma PC - Base orientabile

Monitor Monocromatico 1450 - L. 470.000

Monitor monocromatico BI-SYNC a fosfori "bianco-carta" - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Monitor a colori 1802 - L. 445.000

Monitor a colori - Turbo 14" - Collegabile a C64, C128, C128D

Monitor monocromatico 1900 - L. 199.000

Monitor monocromatico a fosfori verdi - Turbo 12" antiriflesso - Ingresso videocomposito - Compatibile con tutta la gamma Commodore

Monitor a colori 1950 - L. 1.280.000

Monitor a colori BI-SYNC alta risoluzione - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Stampante MPS 1230 - L. 465.000

Stampante a matrice di punti - Testina a 9 aghi - 120 cps - Bidirezionale - 80 colonne - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

MPS 1230R - L. 19.000

Nastro per stampante

Stampante MPS 1500C - L. 495.000

Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia parallela Centronics - Compatibile con la gamma Amiga e PC

MPS1500R - L. 37.000

Nastro a colori per stampante

MPS1500R - L.37.000

Nastro a colori per stampante

Stampante MPS 1550C - L. 575.000

Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

I COMMODORE POINT

LOMBARDIA

Milano
- AL RISPARMIO - V.LE MONZA 204
- ECS - VIA MONTEGANI 11
- BRAHA A. - VIA PIER CAPPONI 5
- E.D.S. - C.SO PORTA TICINESE 4
- FAREF - VIA A. VOLTA 21
- FLOPPERIA - V.LE MONTENERO 31
- GBC - VIA CANTONI 7 - VIA PETRELLA 6
- GIGLIONI - V.LE LUIGI STURZO 45
- L'UFFICIO 2000 - VIA RIPAMONTI 213
- LOGITEK - VIA GOLGI 60
- LU - MEN - VIA SANTA MONICA 3
- MARCUCCI - VIA F.LLI BRONZETTI 37
- MELCHIONI - VIA P. COLLETTA 37
- MESSAGGERIE MUSICALI - GALLERIA DEL CORSO 2
- NEWEL - VIA MAC MAHON 75
- PANCOMMERZ ITALIA - VIA PADOVA 1
- SUPERGAMES - VIA VITRUVIO 38
- 68000 E DINTORNI - VIA WASHINGTON 91

Provincia di Milano
- GINO FERRARI CENTRO HI-FI - VIA MADRE CABRINI 44 - S. ANG. LODIGIANO
- F.LLI GALIMBERTI - VIA NAZIONALE DEI GIOVI 28/36 - BARLASSINA
- TECNOLUX - VIA PIETRO NENNI 5 - BERNATE TICINO
- OGGIONI & C. - VIA DANTE CESANA 27 - CARATE BRIANZA
- AL RISPARMIO - VIA U. GIORDANO 57 - CINISELLO BALSAMO
- GBC - V.LE MATTEOTTI 66 - CINISELLO BALSAMO
- CASA DELLA MUSICA - VIA INDIPENDENZA 21 - COLOGNO MONZESE
- PENATI - VIA VERDI 28/30 - CORBETTA
- EPM SYSTEM - V.LE ITALIA 12 - CORSICO
- P.G. OSTELLARI - VIA MILANO 300 - DESIO
- CENTRO COMPUTER PANDOLFI - VIA CORRIDONI 18 - LEGNANO
- COMPUTEAM - VIA VECELLIO 41 - LISSONE
- M.B.M. - C.SO ROMA 112 - LODI
- L'AMICO DEL COMPUTER - VIA CASTELLINI 27 - MELEGNANO
- BIT 84 - VIA ITALIA 4 - MONZA
- IL CURSORE - VIA CAMPO DEI FIORI 35 - NOVATE MIL.
- I.C.O. - VIA DEI TIGLI 14 - OPERA
- R & C ELGRA - VIA SAN MARTINO 13 - PALAZZOLO MIL.
- ESSEGIEMME SISTEMI SAS - VIA DE AMICIS 24 - RHO
- TECNO - CENTRO - VIA BARACCA 2 - SEREGNO
- NIWA HARD&SOFT - VIA B. BUOZZI 94 - SESTO SAN GIOV.
- COMPUTER SHOP - VIA CONFALONIERI 35 - VILLASANTA
- ACTE - VIA B. CREMIGNANI 13 - VIMERCATE
- IL COMPUTER SERVICE SHOP - VIA PADANA SUPERIORE 197 - VIMODRONE

Bergamo
- D.R.B. - VIA BORGO PALAZZO 65
- TINTORI ENRICO &C. - VIA BROSETA 1
- VIDEO IMMAGINE - VIA CARDUCCI c/o CITTA' DI MERCATO

Provincia di Bergamo
BERTULEZZI GIOVANNI - VIA FANTONI 48 - ALZANO LOMBARDO
- COMPUTER SHOP - VIA VITTORIO VENETO 9 - CAPRIATE SAN GERVASIO
- B.M.R. - VIA BUTTARO 4/T - DALMINE
- MEGABYTE 2 - VIA ROMA 61/A - GRUMELLO
- OTTICO OPTOMETRISTA ROVETTA - P.ZZA GARIBALDI 6 - LOVERE
- COMPUTER POINT - VIA LANTIERI 52 - SARNICO
- A.B. INFORMATICA - STRADA STATALE CREMASCA 66 - URGNANO

Brescia
- MASTER INFORMATICA - VIA F.LLI UGONI 10/B

PROVINCIA DI BRESCIA
- MISTER BIT - VIA MAZZINI 70 - BRENO
- CAVALLI PIETRO - VIA 10 GIORNATE 14 BIS - CASTREZZATO
- VIETTI GIUSEPPE - VIA MILANO 1/B - CHIARI
- MEGABYTE - P.ZZA MALUEZZI 14 - DESENZANO DEL GARDA
- BARESI RINO &C. - VIA XX SETTEMBRE 7 - GHEDI
- INFO CAM - VIA PROVINCIALE 3 - GRATACASOLO
- "PAC-LAND" di GARDONI - CENTRO COM.le - LA CASA DI MARGHERITA D'ESTE - VIA GIORGIONI 21

Como
- IL COMPUTER - VIA INDIPENDENZA 90
- 2M ELETTRONICA - VIA SACCO 3

Provincia di Como
- ELTRON - VIA IV NOVEMBRE 1 - BARZANO'
- DATA FOUND - VIA A. VOLTA 4 - ERBA
- CIMA ELETTRONICA - VIA L. DA VINCI 7 - LECCO
- FUMAGALLI - VIA CAIROLI 48 - LECCO
- RIGHI ELETTRONICA - VIA G. LEOPARDI 26 - OLGIATE COMASCO

Cremona
- MONDO COMPUTER - VIA GIUSEPPINA 11/B
- PRISMA - VIA BUOSO DA DOVARA 8
- TELCO - P.ZZA MARCONI 2/A

Provincia di Cremona
- ELCOM - VIA IV NOVEMBRE 56/58 - CREMA
- EUROELETTRONICA - VIA XX SETTEMBRE 92/A - CREMA

Mantova
- COMPUTER CANOSSA - GAL. FERRI 7
- 32 BIT - VIA C. BATTISTI 14
- ELET. di BASSO - V.LE RISORGIMENTO 69

Provincia di Mantova
- CLICK - ON COMPUTER - S.S. GOITESE 168 - GOITO

Pavia
- POLIWARE - C.SO C. ALBERTO 76
- SENNA GIANFRANCO - VIA CALCHI 5

Provincia di Pavia
- A. FERRARI - C.SO CAVOUR 57 - MORTARA
- LOGICA MAINT - V.LE M.TE GRAPPA 32 - VIGEVANO
- M. VISENTIN - C.SO V. EMANUELE 76 - VIGEVANO

Sondrio
- CIPOLLA MAURO - VIA TREMOGGE 25

Provincia di Sondrio
- FOTONOVA - VIA VALERIANA 1 - S.PIETRO DI BERBENNO

Varese
- ELLE - EFFE - VIA GOLDONI 35
- IL C.TRO ELET. - VIA MORAZZONE 2
- SUPERGAMES - VIA CARROBBIO 13

Provincia di Varese
- BUSTO BIT - VIA GAVINANA 17 - BUSTO A.
- MASTER PIX - VIA S.MICHELE 3 - BUSTO A.
- PUNTO UFFICIO - VIA R.SANZIO 8 - GALLARATE
- GRANDI MAGAZZINI BOSSI - VIA CLERICI 196 - GERENZANO
- J.A.C. - C.so MATTEOTTI 38 - SESTO C.

PIEMONTE

Alessandria
- BIT MICRO - VIA MAZZINI 102
- SERV. INFOR. - VIA ALESSANDRO III 47

Provincia di Alessandria
- SONY ITALIANA - VIA G. MANARA 7 - CASALE MONFERRATO
- SGE ELETTRONICA - VIA BANDELLO 19 - TORTONA
- COMPUTER TEMPLE - VIA F. CAVALLOTTI 13 - VALENZA

Asti
- ASTI GAMES - C.SO ALFIERI 26
- RECORD - C.SO ALFIERI 166/3 (Galleria Argenta)

Cuneo
- ROSSI COMPUTERS - C.SO NIZZA 42

Provincia di Cuneo
- PUNTO BIT - C.SO LANGHE 26/C - ALBA
- BOSETTI - VIA ROMA 149 - FOSSANO
- COMPUTERLAND - VIA MAZZINI 30/32 - SALUZZO

Novara
- PROGRAMMA 3 - V.LE BUONARROTI 8
- PUNTO VIDEO - C.so RISORGIMENTO 39/B

Provincia di Novara
- COMPUTER - VIA MONTE ZEDA 4 - ARONA
- ALL COMPUTER - C.SO GARIBALDI 106 - BORGOMANERO
- S.P.A. - C.SO DISSEGNA 21/BIS - DOMODOSSOLA
- ELLIOTT COMPUTER SHOP - VIA DON MINZONI 32 - INTRA
- TRISCONI VALERIA - VIA MAZZINI 90 - OMEGNA

Torino
- ABA ELETTRONICA - VIA C. FOSSATI 5/P
- ALEX COMPUTER E GIOCHI - C.SO FRANCIA 333/4
- COMPUTER HOME - VIA SAN DONATO 46/D
- COMPUTING NEW - VIA M. POLO 40/E
- C.D.M. ELETTR. - VIA MAROCHETTI 17
- DE BUG - C.SO V. EMANUELE II 22
- DESME UNIVERSAL - VIA S.SECONDO 95
- FDS ALTERIO - VIA BORGARO 86/D
- IL COMPUTER - VIA N. FABRIZI 126
- MICRONTEL - C.SO D. degli ABRUZZI 28
- PLAY GAMES SHOP - VIA C. ALBERTO 39/E
- RADIO TV MIRAFIORI - C.SO UNIONE SOVIETICA 381
- SMT ELETTRONICA - VIA BIBIANA 83/bis

Provincia di Torino
- PAUL E CHICO VIDEOSOUND - VIA V.EMANUELE 52 - CHIERI
- BIT INFORMATICA - VIA V. EMANUELE 154 - CIRIE'
- HI - FI CLUB - C.SO FRANCIA 92C - COLLEGNO
- MISTER PERSONAL - VIA CATTANEO 52 - FAVRIA
- I.C.S. - VIA TORINO 73 - IVREA
- DAG - VIA I MAGGIO 40 - LUSERNA S. GIOVANNI
- EUREX - C.SO INDIPENDENZA 5 - RIVAROLO CANAVESE
- DIAM INFORMATICA - C.SO FRANCIA 146 bis - RIVOLI
- FULLINFORMATICA - VIA V.VENETO 25 - RIVOLI
- GAMMA COMPUTER - VIA CAVOUR 3A-3B - SET.TORINESE

Vercelli
- ELETTROGAMMA - C.SO BORMIDA 27 ang. V.Montanara
- ELETTRONICA - STRADA TORINO 15

Provincia di Vercelli
- C.S.I. TEOREMA - VIA LOSANA 9 - BIELLA
- SIGEST - VIA BERTODANO 8 - BIELLA
- REMONDINO FRANCO - VIA ROMA 5 - BORGOSESIA
- FOTOSTUDIO TREVISAN - VIA XXV APRILE 24/B - COSSATO
- STUDIO FOTOGRAFICO IMARISIO - P.ZZA M. LIBERTA' 7 - TRINO

VENETO

Belluno
- UP TO DATE - VIA V. VENETO 43

Provincia di Belluno
- GUERRA COMPUTERS - V.LE MAZZINI 10/A - FELTRE

Padova
- BIT SHOP - VIA CAIROLI 11
- COMPUMANIA - VIA T. CAMPOSANPIERO 37
- D.P.R. DE PRATO R. - V.LO LOMBARDO 4
- G.F. MARCATO - VIA MADONNA DELLA SALUTE 51/53
- SARTO COMPUTER - VIA ARMISTIZIO 79

Provincia di Padova
- COMPUTER SERVICE - BORGO TREVISO 150 - CITTADELLA

Treviso
- BIT 2000 - VIA BRANDOLINI D'ADDA 14
- GUERRA EGIDIO & C. - V.LE CAIROLI 95

Provincia di Treviso
- DE MARIN COMPUTERS - VIA MATTEOTTI 142 - CONEGLIANO
- SIDESTREET - VIA SALVO D'ACQUISTO 8 - MONTEBELLUNA
- FALCON ELETTROAUDIOVIDEO - VIA TERRAGGIO 116 - PREGANZIOL

Venezia
- GUERRA EGIDIO & C. - VIA BISSUOLA 20/A - MESTRE
- TELERADIO FUGA - SAN MARCO 3457

Provincia di Venezia
- GUERRA EGIDIO & C. - VIA VIZZOTTO 29 - SAN DONA' DI PIAVE
- REBEL - VIA F. CRISPI 10 - SAN DONA' DI PIAVE

Verona
- CASA DELLA RADIO - VIA CAIROLI 10
- TELESAT - VIA VASCO DE GAMA 8

Provincia di Verona
- UBER - CP 0363(RAG SOC. DERTA) - VIA MASCAGNI 31 - CASTE'. D'AZZANO
- FERRARIN - VIA DEI MASSARI 10 - LEGNAGO
- COMPUTERS CENTER - VIA CANTORE 26 - VILLAFRANCA

Vicenza
- ELET. BISELLO - V.LE TRIESTE 427/429
- SCALCHI MARKET - VIA C.A. BALBI 139

Provincia di Vicenza
- SCHIAVOTTO - VIA ZANELLA 21 - CAVAZZALE
- GUERRA E. & C. - V.LE DELLE INDUSTRIE - MONTECCHIO MAGGIORE

FRIULI VENEZIA GIULIA

Gorizia
- E.CO. ELETTRONICA - VIA F.LLI COSSAR 23

Trieste
- AVANZO GIACOMO - P.ZZA CAVANA 7
- COMPUTER SHOP - VIA P. RETI 6
- COMPUTIGI - VIA XX SETTEMBRE 51
- CTI - VIA PASCOLI 4

Udine
- MOFERT 2 - VIA LEOPARDI 21
- R.T. SISTEM UDINE - VIA L. DA VINCI 99

Provincia di Udine
- IL PUNTO ELETTRONICO - VIA VENDRAMIN 184 - LATISANA
- IDRENO MATTIUSSI &C. - VIA LICINIANA 58 - MARTIGNACCO

TRENTINO ALTO ADIGE

Bolzano
- COMPUTER POINT - VIA ROMA 82/A
- MATTEUCCI PRESTIGE - VIA MUSEO 54

Provincia di Bolzano
- RADIO MAIR-ELECTRO - VIA CENTRALE 70 - BRUNICO
- ELECTRO RADIO HENDRICH - VIA DELLE CORSE 106 - MERANO
- ERICH KONTSCHIEDER - PORTICI 313 - MERANO
- ELECTRO TAPPEINER - P.ZZA PRINCIPALE 90 - SILANDRO

Trento
- CRONST - VIA G. GALILEI 25

Provincia di Trento

• AL RISPARMIO - C.SO VERONA 138 - ROVERETO

LIGURIA
Genova
• ABM COMPUTER - P.ZZA DE FERRARI 24 rosso
• CAPRIOTTI G. - IA MAMIANI 4r - SAMPIERDARENA
• C.tro ELET. - VIA CHIARAVAGNA 10 R - VIA SESTRI 69R
• COM.le SOTTORIPA - VIA SOTTORIPA 115/117
• FOTOMONDIAL - VIA DEL CAMPO 3-5-9-11-13 r
• LA NASCENTE - VIA SAN LUCA 4/1
• PLAY TIME - VIA GRAMSCI 3/5/7 rosso
• RAPPR-EL - VIA BORGORATTI 23 R
Imperia
• CASTELLINO - VIA BELGRANO 44
Provincia di Imperia
• CENTRO HI-FI VIDEO - VIA DELLA REPUBBLICA 38 -SANREMO
• CASTELLINO - VIA GENOVA 48 - VENTIMIGLIA
La Spezia
• I.L. ELETTRONICA - VIA V. VENETO 123
Provincia di La Spezia
• I.L. ELETTRONICA - VIA AURELIA 299 - FORNOLA DI VEZZANO
Savona
• CASTELLINO - C.SO TARDY E BENECH 101
Provincia di Savona
• CELESIA ENZA - VIA GARIBALDI 144 - LOANO

EMILIA
Bologna
• EUROELETTRICA - VIA RANZANI 13/2
• MINNELLA ALTA FEDELTA - VIA MAZZINI 146/2
• MORINI & FEDERICI - VIA MARCONI 28/C
• STERLINO - VIA MURRI 73/75
Provincia di Bologna
• S.C. COMPUTERS - VIA E. FERMI 4 - CASTEL SAN PIETRO
• S.P.E. INFORMATICA - VIA DI MEZZO PONENTE 385 - CREVALCORE
• ARCHIMEDE SISTEMI - VIA EMILIA 124 - S. LAZZARO DI SAVENA
Modena
• CO - EL - VIA CESARI 7
• ORSA MAGGIORE - P.ZZA MATTEOTTI 20
• VIDEO VAL WILLY COMPUTERS - VIA CANALETTO 223
Provincia di Modena
• NEW MEDIA SYSTEM - VIA ROMA 281 - SOLIERA
Parma
• BABARELLI G. - VIA B. PARENTE 14/A/B
Provincia di Parma
• PONGOLINI - VIA CAVOUR 32 - FIDENZA
Piacenza
• COMPUTER LINE - VIA G. CARDUCCI 4
• DELTA COMPUTER - VIA M. DELLA RESISTENZA 15/G

TEGGIO EMILIA
• COMPUTERLINE - VIA SAN ROCCO 10/C
• POOL SHOP - VIA EMILIA S. STEFANO 9/C
Provincia di Reggio Emilia
• MACCHIONI - VIA STATALE 467 - CASALGRANDE
ROMAGNA
Ferrara
• BUSINESS POINT - VIA CARLO MAYER 85
Forlì
• COMPUTER VIDEO CENTER - VIA CAMPO DI MARTE 122
Provincia di Forlì
• TOP BIT - VIA VENETO 12 - FORLIMPOPOLI
• COMPUTER HOUSE - V.LE TRIPOLI 193/D - RIMINI
• EASY COMPUTER - VIA LAGOMAGGIO 50 - RIMINI

REPUBBLICA S. MARINO
Ravenna
• COMPUTER HOUSE - VIA TRIESTE 134
Provincia di Ravenna
• ARGNANI - P.ZZA DELLA LIBERTA' 5/A - FAENZA
• ELECTRON INFORMATICA - VIA F.LLI CORTESI 17 - LUGO
• P.L.Z. INFORMATICA - P.ZZA SERCOGNANI 6 - FAENZA

TOSCANA
Arezzo
• DELTA SYSTEM - VIA PIAVE 13
Firenze
• ATEMA - VIA BENEDETTO MARCELLO 1a-1b
• ELETTRONICA CENTOSTELLE - VIA CENTO STELLE 5/a-b
• HELP COMPUTER - VIA DEGLI ARTISTI 15-A
• TELEINFORMATICA TOSCANA -VIA BRONZINO 36
Provincia di Firenze
• WAR GAMES - VIA R. SANZIO 126/A - EMPOLI
• NEW EVM COMPUTER - VIA DEGLI INNOCENTI 2 - FIGLINE VALDARNO
• C.tro INFOR. - VIA ZNOJMO 41 - PONTASSIEVE
• COSCI F.LLI - VIA ROMA 26 - PRATO
• BARBAGLI C. ELET. - VIA F. BONI 80 - PRATO
Grosseto
• COMPUTER SERVICE - VIA DELL'UNIONE 7
Livorno
• ETA BETA - VIA SAN FRANCESCO 30
• FUTURA 2 - VIA CAMBINI 19
Provincia di Livorno
• PUNTO ROSSO - VIA BARONTINI 28 - PIOMBINO
Provincia di Lucca
• IL COMPUTER - V.LE COLOMBO 216 - LIDO DI CAMAIORE
• SANTI VITTORIO - VIA ROMA 23 - S. ROMANO GARFAGNANA
• TOP GAMES - VIA S. ANDREA 122 - VIAREGGIO
Massa
• EURO COMPUTER - P.ZZA G. BERTAGNINI 4
Carrara
• RADIO LUCONI - VIA ROMA 24/B
Pisa
• ELECTRONIC SERVICE - VIA DELLA VECCHIA TRANVIA 10
• PUCCINI S. - CP 1199 (RAG.SOC. MAREXI - VIA C.CAMMEO 64
• TONY HI-FI - VIA CARDUCCI
Provincia di Pisa
• M.C. INFORMATICA - VIA DEL CHIESINO 4 - PONTEDERA (PI)
Pistoia
• ELECTRONIC SHOP - VIA DEGLI SCALZI 3
Provincia di Pistoia
• ZANNI &C. - C.SO ROMA 45 - MONTECATINI T.

Siena
• R. BROGI - P.ZZA GRAMSCI 28
• VIDEO MOVIE - VIA GARIBALDI 17
Provincia di Siena
• ELETTRONICA di BIFOLCHI - VIA DI GRACCIANO NEL CORSO 111 - MONTEPULCIANO

LAZIO
• CENTRO INF. - D.R.R. srl - TEL. 06-5565672

UMBRIA
Perugia
• MIGLIORATI - VIA S. ERCOLANO 3-10
Provincia di Perugia
• COMPUTER STUDIO'S - VIA IV NOVEMBRE 18/A - BASTIA UMBRA
• WARE - VIA DEI CASCERI 31 - CITTA'DI CASTELLO
Terni
• CGS SOFTWARE HOUSE - VIA DONIZETTI 71/A

BASILICATA
Matera
• G. GAUDIANO ELECTRONICS - VIA ROMA ang. XX SETTEMBRE 1

PUGLIA
Bari
• ARTEL - VIA GUIDO D'ORSO 9
• COMPUTER'S ARTS - V.LE MEUCCI 12/B
• PAULICELLI S. & F. - VIA FANELLI 231/C
Provincia di Bari
• F. FAGGELLA - C.SO GARIBALDI 15 - BARLETTA
• G.FAGGELLA - P.ZZA D'ARAGONA 62A - BARLETTA
• LONUZZO G. - VIA NIZZA 21 - CASTELLANA
• TECNOUFF. - VIA RICASOLI 54 - MONOPOLI
• TANGORRA N. - C.SO V.EMANUELE 130/B - TRIGGIANO
Brindisi
• MARANGI E NICCOLI - VIA PROV. SAN VITO 165
Provincia di Brindisi
• MILONE G. - VIA S.F. D'ASSISI 219 - FRANCAVILLA FONTANA
Foggia
• BOTTICELLI G. - VIA SAV POLLICE 2
• E.C.I. COMPUTER - VIA ISONZO 28
• LA TORRE - V.LE MICHELANGELO 185
Provincia di Foggia
• IL DISCOBOLO - VIA T. SOLIS 15 - SAN SEVERO
Lecce
• BIT - VIA 95 REGG.NTO FANTERIA 87/89
Provincia di Lecce
• TECNO UFFICIO - P.ZZA GIOVANNI XXIII 10 - GALLIPOLI
• CEDOK INFORMATICA - VIA UMBERTO I 116 - TRICASE
Taranto
• ELETTROJOLLY C.tro - VIA DE CESARE 13
• TEA - TEC. ELET. AV. - VIA R. ELENA 101

CAMPANIA
Provincia di Avellino
• FLIP FLOP - VIA APPIA 68 - ATRIPALDA
Benevento
• E.CO. INF. - VIA PEPICELLI 21/25
Caserta
• ENTRY POINT - VIA COLOMBO 31
• O.P.C. - VIA G. M. BOSCO 24
Provincia di Caserta
• M.P. COMPUTER - VIA NAPOLI 30 - MADDALONI
• DAMIANO - C.SO V. EMANUELE 23 -ORTA DI ATELLA
• FUSCO B. - VIA NAPOLI 24 - VAIRANO PATERNORA (FRAZ. VAIRANO SCALO)
• LINEA CONTABILE - VIA OSPEDALE 72/76 - SESSA A. (CE)
Napoli
• BABY TOYS - VIA CISTERNA DELL'OLIO 5/BIS
• CASA MUSICALE RUGGIERO - P.ZZA GARIBALDI 74 (INT. STAZ. F.F. S.S.)
• C.tro ELET. CAMPANO - VIA EPOMEO 121
• CI AN - GALLERIA VANVITELLI 32
• CINE NAPOLI - VIA S. LUCIA 93 95
• DARVIN - CALATA SAN MARCO 26
• GIANCAR 2 - P.ZZA GARIBALDI 37
• ODORINO - L.GO LALA 22 A-B
• R 2 - VIA F. CILEA 285
• SAGMAR - VIA S. LUCIA 140
• TOP VIDEO - TOP COMPUTER - VIA S. ANNA DEI LOMBARDI 12
• VIDEOFOTOMARKET - VIA S. BRIGIDA 19
Provincia di Napoli
• ELECTRONIC DAY - VIA DELLE PUGLIE 17 - CASORIA
• TUFANO - S.S. SANNITICA 87 KM 7 - CASORIA
• SOF SUD - V.LE EUROPA 59 - CASTEL/MARE DI STABIA
• ELETTRONICA 2000 - C.SO DURANTE 40 - FRATTAMAGGIORE
• SPADARO - VIA ROMANI 93 - MADONNA DELL'ARCO
• GATEWAY - VIA NAPOLI 68 - MUGNANO
• VISPINI & DI VUOLO - VIA A.ROSSI 4 - POMPEI
• SPY CASH & CARRY - P.ZZA ARENELLA 6/A - NAPOLI
• NUOVA INFORMATICA SHOP - VIA LIBERTA' 185/191 PORTICI
• BASIC COMPUTER - C.SO GARIBALDI 34 - POZZUOLI
• V.C. - C.SO SECONDIGLIANO 562/B - SECONDIGLIANO
• F. ELETTRONICA - VIA SARNO 102 - STRIANO
• TECNO - VIA V. VENETO 48 - TORRE DEL GRECO
Salerno
• COMPUMARKET - VIA BELVEDERE 35
• COMPUTER MARKET - C.SO VITTORIO EMANUELE 23
Provincia di Salerno
• KING COMPUTER - VIA OLEVANO 56 - BATTIPAGLIA
• DIMER POINT - V.LE AMENDOLA 36 - EBOLI
• IACUZIO F. - VIA MUNICIPIO 14 - MERCATO SAN SEVERINO
• COMPUTER SERVICE - VIA L. DA VINCI 81 - SCAFATI

CALABRIA
Catanzaro
• C. & G. COMPUTER - VIA F. ACRI 28
• PAONE S. & F. - VIA F. ACRI 93/99
Provincia di Catanzaro
• COMPUTER HOUSE - VIA BOLOGNA (L.GO OSPEDALE) - CROTONE
• RIOLO F.LLI - VIA VENEZIA 1/7 - CROTONE
• ING. FUSTO S. - C.SO NICOTERA 99 - LAMEZIA TERME
Cosenza
• MAISON DE L'INFORMATIQUE - VIA PASQUALE ROSSI 34/C
• SIRANGELO COMP. - VIA N. PARISIO 25
Provincia di Cosenza
• HI-FI ALFANO G. - VIA BALDACCHINI 109 - AMANTIA
• ELIGIO ANNICCHIARICO &C. - VIA ROMA 21 - CASTROVILLARI
• ALFA COMPUTER - VIA NAZIONALE 341/A - CORIGLIANO SCALO
REGGIO CALABRIA
• CONTROL SYSTEM - VIA S.F DA PAOLA 49 D
• SYSTEM HOU - VIA FIUME ang. PALESTINO 1
Provincia di Reggio Calabria
• COMPUTER SHOP - V.LE MATTEOTTI 36/38 - LOCRI
• PICIEFFE - C.SO F. S. ALESSIO 19 - TAURIANOVA
SICILIA
• CENTRO INF. - ITALSOFT SRL - TEL. 0935-696090

DIMENSIONE
AVVENTURA
JONATHAN
OGNI MESE
IN EDICOLA
JONATHAN
TANZANIA
Safari
in mongolfiera
INDIA
MONTAGNA
SPECIALE
Avventura
nei cieli